N 11

```
VVV         VVV  MMM         MMM     SSSSSSSSSSSS  LLL                 IIIIIIIII     BBBBBBBBBBBB
VVV         VVV  MMM         MMM     SSSSSSSSSSSS  LLL                 IIIIIIIII     BBBBBBBBBBBB
VVV         VVV  MMM         MMM     SSSSSSSSSSSS  LLL                 IIIIIIIII     BBBBBBBBBBBB
VVV         VVV  MMMMMM   MMMMMM  SSS              LLL                    III        BBB         BBB
VVV         VVV  MMMMMM   MMMMMM  SSS              LLL                    III        BBB         BBB
VVV         VVV  MMMMMM   MMMMMM  SSS              LLL                    III        BBB         BBB
VVV         VVV  MMM   MMM   MMM  SSS              LLL                    III        BBB         BBB
VVV         VVV  MMM   MMM   MMM  SSS              LLL                    III        BBB         BBB
VVV         VVV  MMM   MMM   MMM  SSS              LLL                    III        BBB         BBB
VVV         VVV  MMM         MMM     SSSSSSSSS     LLL                    III        BBBBBBBBBBBB
VVV         VVV  MMM         MMM     SSSSSSSSS     LLL                    III        BBBBBBBBBBBB
VVV         VVV  MMM         MMM     SSSSSSSSS     LLL                    III        BBBBBBBBBBBB
VVV         VVV  MMM         MMM              SSS  LLL                    III        BBB         BBB
VVV         VVV  MMM         MMM              SSS  LLL                    III        BBB         BBB
VVV         VVV  MMM         MMM              SSS  LLL                    III        BBB         BBB
   VVV   VVV     MMM         MMM              SSS  LLL                    III        BBB         BBB
   VVV   VVV     MMM         MMM              SSS  LLL                    III        BBB         BBB
   VVV   VVV     MMM         MMM              SSS  LLL                    III        BBB         BBB
      VVV        MMM         MMM  SSSSSSSSSSSS     LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB
      VVV        MMM         MMM  SSSSSSSSSSSS     LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB
      VVV        MMM         MMM  SSSSSSSSSSSS     LLLLLLLLLLLLLLL     IIIIIIIII     BBBBBBBBBBBB
```

```
**FILE**ID**STARDEFAE                                N 3



  SSSSSSSS  TTTTTTTTTT    AAAAAA    RRRRRRRR    DDDDDDDD    EEEEEEEEEE  FFFFFFFFFF    AAAAAA    EEEEEEEEEE
  SSSSSSSS  TTTTTTTTTT    AAAAAA    RRRRRRRR    DDDDDDDD    EEEEEEEEEE  FFFFFFFFFF    AAAAAA    EEEEEEEEEE
SS              TT      AA      AA  RR      RR  DD      DD  EE          FF          AA      AA  EE
SS              TT      AA      AA  RR      RR  DD      DD  EE          FF          AA      AA  EE
SS              TT      AA      AA  RR      RR  DD      DD  EE          FF          AA      AA  EE
SS              TT      AA      AA  RR      RR  DD      DD  EE          FF          AA      AA  EE
  SSSSSS        TT      AA      AA  RRRRRRRR    DD      DD  EEEEEEEE    FFFFFFFF    AA      AA  EEEEEEEE
  SSSSSS        TT      AA      AA  RRRRRRRR    DD      DD  EEEEEEEE    FFFFFFFF    AA      AA  EEEEEEEE
        SS      TT      AAAAAAAAAA  RR  RR      DD      DD  EE          FF          AAAAAAAAAA  EE
        SS      TT      AAAAAAAAAA  RR  RR      DD      DD  EE          FF          AAAAAAAAAA  EE
        SS      TT      AA      AA  RR    RR    DD      DD  EE          FF          AA      AA  EE          ....
        SS      TT      AA      AA  RR    RR    DD      DD  EE          FF          AA      AA  EE          ....
SSSSSSSS        TT      AA      AA  RR      RR  DDDDDDDD    EEEEEEEEEE  FF          AA      AA  EEEEEEEEEE  ....
SSSSSSSS        TT      AA      AA  RR      RR  DDDDDDDD    EEEEEEEEEE  FF          AA      AA  EEEEEEEEEE  ....

  SSSSSSSS  DDDDDDDD    LL
  SSSSSSSS  DDDDDDDD    LL
SS          DD      DD  LL
SS          DD      DD  LL
SS          DD      DD  LL
SS          DD      DD  LL
  SSSSSS    DD      DD  LL
  SSSSSS    DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
SSSSSSSS    DDDDDDDD    LLLLLLLLLL
SSSSSSSS    DDDDDDDD    LLLLLLLLLL
```

---

```
STA

end
end
```

```
{ STARDEFAE.SDL - system user interface definitions
{
{ Version:      'V04-000'
{
{****************************************************************************
{*                                                                          *
{*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
{*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
{*  ALL RIGHTS RESERVED.                                                    *
{*                                                                          *
{*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
{*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
{*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
{*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
{*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
{*  TRANSFERRED.                                                            *
{*                                                                          *
{*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
{*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
{*  CORPORATION.                                                            *
{*                                                                          *
{*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
{*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
{*                                                                          *
{*                                                                          *
{****************************************************************************

{++
{
{ FACILITY:  VAX/VMS System Macro Libraries
{
{ ABSTRACT:
{
{       This file contains the SDL source for all user visible operating
{       system interfaces from A to E.
{
{ ENVIRONMENT:
{
{       n/a
{
{--
{
{
{ AUTHOR:  The VMS Group         CREATION DATE:  1-Aug-1976
{
{ MODIFIED BY:
{
{       V03-085 CWH3085         CW Hobbs                21-AUG-1984
{               Remove {} from comments in DVIDEF because of problems
{               with the PASCAL back end.
{
{       V03-084 CWH3084         CW Hobbs                24-JUL-1984
{               Add spare $DVIDEF codes for shadow contingency.
{
{       V03-083 ROW0391         Ralph O. Weber          18-JUL-1984
{               Add device types for the following CDR50, QDA25, RX31, RX32,
{       and RX18.  Also add port device types where appropriate.
{
{       V03-082 LY0506          Larry Yetto             11-JUL-1984 17:33
{               Add MEDIA_ID item code for GETDVI
{
{       V03-081 LY0504          Larry Yetto             10-JUL-1984 10:20
{               Add MEDIA_NAME and MEDIA_TYPE item codes to $DVIDEF
{
{       V03-080 EAD182          Elliott A. Drayton      26-Jun-1984
{               Add DT$_NQ_3271 and DT$_VD for the DHFC and Vir. Workstation 100
{
{       V03-079 LMP0262         L. Mark Pilant,         26-Jun-1984  12:40
{               Add ACL$S_UNLOCK_ACL to $ACLDEF.
{
{       V03-078 RLRQDA          Robert L. Rappaport     19-Jun-1984
{               Add DT$_ definitions for QDA and BDA.
{
{       V03-077 RAS0300         Ron Schaefer            27-Apr-1984
{               Add DEV$V_NNM device characteristic to DEVCHAR2 to specify
{               that cluster nodenames should be prefixed to the device name.
{
{       V03-076 ACG0415         Andrew C. Goldstein,    10-Apr-1984  11:38
{               Change file name attribute length to 86 bytes
{
{       V03-075 RKS0075         RICK SPITZ              06-APR-1984
{               Add DEVDEF bit to indicate redirected Physical Terminal UCB
{
{       V03-074 HWS0049         Harold Schultz          02-Apr_1984
{               Add CLI$Q_TABLE to SPAWN data structure.
{
{       V03-073 LMP0213         L. Mark Pilant,         24-Mar-1984  11:43
{               Add additional item codes to $ACLDEF for locking and unlocking
{               the object's ACL.
{
{       V03-072 RLRCRX50        Robert L. Rappaport     13-Mar-1984
{               Add DT$_CRX50 to $DCDEF.
{
{       V03-071 EMD0062         Ellen M. Dusseault      8-Mar-1984
{               Add DVI$_TT_DECCRT2 to $DVIDEF.  Add DT$_DHU and
{               DT$_DHV ( terminal controller definitions ) to
{               $DCDEF.
{
{       V03-070 MHB0105         Mark Bramhall           1-Mar-1984
{               Added DVI$_TT_PHYDEVNAM to $DVIDEF.
{
{       V03-069 ROW0315         Ralph O. Weber          27-FEB-1984
{               Add new device types produced by revised MSCP specification to
{               $DCDEF.
{
{       V03-068 CWH3068         CW Hobbs                24-Feb-1984
{               Add codes to $DVIDEF to support dual-path and shadow
{               set characteristics
{
{       V03-067 HH0003          Hai Huang               16-Feb-1984
{               Add DMT$M_ABORT, DMT$M_CLUSTER to $DMTDEF.
{
```

```
{       V03-066 LMP0186         L. Mark Pilant,         31-Jan-1984  11:25
{               Add additional items to the $CHANGE_ACL service item list.
{
{       V03-065 ROW0291         Ralph O. Weber          29-JAN-1984
{               Add DEV$M_SRV, a device characteristics bit definition for
{               DEVCHAR2 which when set indicates that the device is being
{               served to the VAXcluster via the MSCP server.  Add DT$_FD1,
{               DT$_FD2, DT$_FD3, ...., DT$_FD8.  These are eight foreign disk
{               device types.  They provide the basic mechanism for serving
{               foreign disks to the VAXcluster via the MSCP server.
{
{       V03-064 MIR0300         Michael I. Rosenblum    10-Jan-1984
{               add definition for BTS terminals and make a note
{               that the negitive terminal device types are reserved
{               for the RTL foreign terminal support.
{               Move the definition for VT200 out of the negitive range
{
{       V03-063 ROW0276         Ralph O. Weber          7-JAN-1984
{               Add DEV$M_SSM, shadow set member, to $DEVDEF.
{
{       V03-062 MMD0213         Meg Dumont,     13-Dec-1983  17:17
{               Add the constant ATR$C_BUFFER_OFFSET to ATRDEF to return
{               the value of buffer offset for magnetic tape files.
{
{       V03-061 LMP0177         L. Mark Pilant,         7-Dec-1983  12:53
{               Add ATR$x_CLASS_MASK to $ATRDEF to set/get the classification
{               mask from the file header.  Also, add a new structure $ACLDEF,
{               to define the item codes used in the $SETACL system service.
{
{       V03-060 LMP0175         L. Mark Pilant,         2-Dec-1983  10:30
{               Add support for the RMS journal-ID fields in $ACEDEF.
{
{       V03-059 EAD0101         Elliott A. Drayton      30-nov-1983
{               Add two realtime devices FP-FEPCM and TK-FCM.
{
{       V03-058 ROW0251         Ralph O. Weber          11-NOV-1983
{               Add DEV$M_MSCP, a device characteristics bit which indicates
{               that the device is accessed using the MSCP protocol.  This bit
{               is being added over some objections to the effect that MSCP
{               and non-MSCP devices are (or should be) sufficiently alike
{               that no need to differentiate between them exists.  Generally
{               speaking, this is true and care must be taken to avoid relying
{               on the DEV$M_MSCP bit when other means of differentiating
{               between devices are superior.  However, the number of cases
{               where the DEV$M_MSCP bit will be the only correct decision
{               mechanism is growing, and the bit is truely needed.
{
{       V03-057 LMP0170         L. Mark Pilant,         11-Nov-1983  15:48
{               Add ACE$M_NOPROPAGATE and ACE$M_UNIQUE ACE type modifiers.
{
{       V03-056 TMK0001         Todd M. Katz            11-Oct-1983
{               Re-define DT$_LSIUNA to be DT$_DELUA, and add DT$_DEUNA
{               and assign it the same value as DT$_UNA11.
{
{       V03-055 EAD0087         Elliott A.Drayton       10-SEP-1983
{               Add definitions for WORKSTATION class and three types.
```

```
{
{       V03-054 ROW0225         Ralph O. Weber          22-SEP-1983
{               Add DEV$V_2P which indicates that two paths are known to a
{               device, when set.
{
{       V03-053 PCG0003         Peter C. George         15-Sep-1983
{               Add CLI$B_VERSION to spawn data structure.
{               Add CLI$K_SPAWN_VERSION.
{               Change RC25 port definition to LESI.
{
{       V03-052 ACG0354         Andrew C. Goldstein,    13-Sep-1983  19:32
{               Change CHP$_READWRITE item to CHP$_FLAGS, add READALL flag;
{               also remove ATR$C_ACCESS_MASK item.
{
{       V03-051 ROW0223         Ralph O. Weber          13-SEP-1983
{               Add DTs for RC25 and RCF25, the AZTEC.  Make the values equal
{               to those of the RZ01 and RZF01, the old names for AZTEC.
{
{       V03-050 KFH0005         Ken Henderson           10 Sep 1983
{               Add DVI$_VOLSETMEM and DVI$_DEVLOCKNAM
{               to $DVIDEF.
{
{       V03-049 KFH0004         Ken Henderson           22 Aug 1983
{               Add new devdepnd2 bits to $DVIDEF.
{               Prefix $DVIDEF's devdepend/devdepnd2 bits with "TT_".
{               Remove items from $DVIDEF that were bits defined in SYSDEF.
{               Add DMZ32 to $DCDEF for Mike Rosenblum.
{
{       V03-048 TCM0003         Trudy C. Matthews       28-Jul-1983
{               Add DVI$_ALLDEVNAM code to $DVIDEF.
{
{       V03-047 JLV0278         Jake VanNoy             27-JUL-1983
{               Add BRK$C_DCL and change BRK$M_REFRSH to BRK$M_NOREFRESH.
{
{       V03-046 WMC0046         Wayne Cardoza           27-Jul-1983
{               New flag bits for checkpoint calls.
{
{       V03-045 RNG0045         Rod Gamache             27-Jul-1983
{               Add device types for new DECnet-VAX hardware.
{
{       V03-044 PCG0002         Peter George            27-Jun-1983
{               Add new CLI$ fields for SPAWN callback to DCL.
{               Add new CLI$ fields for enhanced logical name callbacks.
{               Add UK_KTC32 and TQ_BTS to $DCDEF.
{
{       V03-043 RLRDPATH        Robert L. Rappaport     23-Jun-1983
{               Add DEV$M_CDP, meaning local DISK or TAPE is also visible
{               thru an MSCP Server on another node.  This bit means
{               that the device has two UCB's in the local I/O database,
{               one the normal local UCB, the other a Disk (or Tape) Class
{               Driver UCB.  These two UCB's point at each other thru
{               field UCB$L_DP_ALTUCB.
{
{       V03-042 LMP0124         L. Mark Pilant,         22-Jun-1983  9:44
{               Change the OWNER access definition to CONTROL.
{
```

```
STA

agg

end

end_
```

```
{       V03-041 WMC0041         Wayne Cardoza           21-Jun-1983
{               Add CHKPNT macro
{
{       V04-040 TCM0002         Trudy C. Matthews       20-Jun-1983
{               Add DVI$_LOCKID item code to $DVIDEF.
{
{       V03-039 LMP0120         L. Mark Pilant,         16-Jun-1983  9:07
{               Fix the sizes of the ACP ACL attributes.
{
{       V03-038 JLV0270         Jake VanNoy             14-JUN-1983
{               Add two bits to DEVDEF - RTT and DET. Add more to
{               $BRKDEF. Add new realtime device to DCDEF.
{
{       V03-037 LMP0116         L. Mark Pilant,         19-May-1983  9:29
{               Add support for directory default protection ACEs.
{
{       V03-036 RSH0014         R. Scott Hanna          29-Apr-1983
{               Added the file Access Rights Mask bit definition
{               macro $ARMDEF.
{
{       V03-035 KFH0003         Ken Henderson           29 Apr 1983
{               Added FULLDEVNAM to $DVIDEF.
{
{       V03-034 JLV0247         Jake VanNoy             29-APR-1983
{               Add $BRKDEF, inputs to $BRKTHRU system service.
{
{       V03-033 TCM0001         Trudy C. Matthews       28-Apr-1983
{               Add DEV$V_CLU bit to $DEVDEF.  This bit signals that
{               the device is available cluster-wide.
{
{       V03-032 MLJ0112         Martin L. Jack          27-Apr-1983
{               Delete $DJIDEF (superseded by new structure, in LIB)
{               and $DJTDEF (obsoleted).
{
{       V03-031 LMP0101         L. Mark Pilant,         14-Apr-1983  16:00
{               Add ATR$x_FILE_SPEC to translate a file-ID to a full file
{               specification.
{
{       V03-030 ROW0171         Ralph O. Weber          12-APR-1983
{               Add GETDVI item code DEVCHAR2 for the second device
{               characteristics longword, UCB$L_DEVCHAR2.  Although this
{               longword immediately follows UCB$L_DEVCHAR in the UCB, its
{               value is returned seperately.  This conforms to the precident
{               set by UCB$L_DEVDEPEND2 and prevents GETDVI from returning any
{               quadword values.
{
{       V03-029 LMP0098         L. Mark Pilant,         8-Apr-1983  12:48
{               Add a new attribute for specifying the access mode for an
{               attribute.  Also, add a new journal-id ACE type.
{
{       V03-028 WMC0028         Wayne Cardoza           31-Mar-1983
{               Remove CRPROCDEF.
{
{       V03-027 LMP0093         L. Mark Pilant,         28-Mar-1983  8:34
{               Add CHP$_ACCESSRIGHTS as a mechanism for returning the
{               access rights mask being checked.
```

```
STA

modu

/*+
/* 
/*-

aggr




end

end_
```

```
{
{       V03-026 ROW0172         Ralph O. Weber          25-MAR-1983
{               Add unique mailbox device types for shared memory mailboxes,
{               DT$_SHRMBX, and the null device, DT$_NULL.
{
{       V03-025 STJ3076         Steven T. Jeffreys,     25-Mar-1983
{               - Added support in $ATRDEF for file high-water marks.
{               - For LMP added the following to $ATRDEF:
{                       ATR$x_ACCESS_MASK
{                       ATR$x_PRIVS_USED
{                       ATR$x_MATCHING_ACE
{
{       V03-024 RLRUQPORT       Robert L. Rappaport     18-Mar-1983
{               Add DT$_ definitions for UQPORT devices; UDA50A, AZTEC,
{               TU81, RDRX.
{
{       V03-023 LMP0087         L. Mark Pilant,         11-Mar-1983  10:57
{               Modify the $ACEDEF structure to allow for hidden and/or
{               protected ACEs.  Also, add a new INFO ACE type.
{
{       V03-022 KFH0002         Ken Henderson            9 Mar 1983
{               Added STS and DEVSTS definitions
{               plus their bit definitions to $DVIDEF.
{
{       V03-021 LMP0082         L. Mark Pilant,         25-Feb-1983  9:24
{               Add SUCCESS and FAILURE flags, remove the ALLGROUP and
{               ALLMEMBER flags, and add some offsets for the various
{               journal names.  Also add interface to the check protection
{               system service $CHPDEF.
{
{       V03-020 KFH0001         Ken Henderson           24 Feb 1983
{               Add DEVCHAR and DEVDEPEND(2) bit
{               definitions to $DVIDEF.
{
{       V03-019 RLRRDRX         Robert L. Rappaport     9-Feb-1983
{               Add RD51 and RX50 definitions in $DTDEF.
{
{       V03-018 ACG0307         Andrew C. Goldstein,    18-Jan-1983  11:34
{               Remove password ACE type
{
{       V03-017 LMPbbbb         L. Mark Pilant,         18-Jan-1983  10:36
{               Move the definition of the ACE overhead area to the correct
{               location.  The length was off by 4 bytes.
{
{       V03-016 ACG0307         Andrew C. Goldstein,    30-Dec-1982  16:26
{               Add passwords and reserved area to ACE's, add
{               classification mask block
{
{       V03-015 STJ3043         Steven T. Jeffreys,     16-Dec-1982
{               Add the $ERADEF macro definiton.
{
{       V03-014 ACG0303         Andrew C. Goldstein,    9-Dec-1982  16:05
{               Add FILL attribute to extraneous field names
{
{       V03-013 LMP0062         L. Mark Pilant,         9-Dec-1982  14:55
{               Add new ACE type codes for security audit journals and
```

---

```
STA

modu
/*
/*
/*
/*


agg
```

```
{               security alarms.  Also, Ace types for BI, AI, and AT RMS
{               journal names.
{
{       V03-012 PCG0001         Peter George            22-Nov-1982
{               Add a new CLINT service code.
{
{       V03-011 LMP0054         L. Mark Pilant,         25-Oct-1982  16:55
{               Add a new attribute to get the length of a file's ACL.
{
{       V03-010 WMC0002         Wayne Cardoza           15-Oct-1982
{               Add another CREPRC code.
{
{       V03-009 WMC0001         Wayne Cardoza           04-Oct-1982
{               Add CREPRC item list definitions.
{
{       V03-008 EAD0001         Elliott A. Drayton      21-Sep-1982
{               Added DT$_XP_PCL11B.
{
{       V03-007 RLR0001         Robert L. Rappaport     22-July-1982
{               Added DT$_TA78, _TU80, _TU81, and _TA81.
{
{       V03-006 JWH0001         Jeffrey W. Horn         02-Jul-1982
{               Add Journaling Names type to Access Control Entry.
{
{       V03-005 LMP0036         L. Mark Pilant,         29-Jun-1982  10:40
{               Add the structure definition for the Access Control Entry.
{               Also, add the necessary attributes to support the ACL editing
{               functions in the ACP.
{
{       V03-004 JSV006          Joost Verhofstad        10-Jun-1982
{               Added DT$_CLJNL.
{
{
{----
```

```
{
{ ACCOUNTING AND TERMINATION MESSAGE FORMAT
{ THIS IS THE STRUCTURE OF THE MESSAGE SENT TO THE TERMINATION MAILBOX
{ AND TO THE JOB CONTROLLER.
{
module $ACCDEF;


aggregate ACCDEF structure prefix ACC$;
    MSGTYP word unsigned;                               /* MESSAGE TYPE CODE
    MSGSIZ word unsigned;                               /* LENGTH OF DATA MESSAGE (VALID ONLY IN ACCLOG)
    FINALSTS longword unsigned;                         /* FINAL EXIT STATUS
    PID longword unsigned;                              /* PROCESS ID
    JOBID longword unsigned;                            /* JOB IDENTIFICATION (VALID ONLY IN ACCLOG)
    TERMTIME quadword unsigned;                         /* TERMINATION TIME (100NS UNITS)
    ACCOUNT character length 8;                         /* ACCOUNT NAME STRING (BLANK FILLED)
    USERNAME character length 12;                       /* USER NAME STRING (BLANK FILLED)
    CPUTIM longword unsigned;                           /* CPUTIM IN 10MS UNITS
    PAGEFLTS longword unsigned;                         /* TOTAL PAGE FAULTS
    PGFLPEAK longword unsigned;                         /* PEAK PAGING FILE USAGE
    WSPEAK longword unsigned;                           /* PEAK WORKING SET SIZE
    BIOCNT longword unsigned;                           /* COUNT OF BUFFERED I/O OPERATIONS
    DIOCNT longword unsigned;                           /* COUNT OF DIRECT I/O OPERATIONS
    VOLUMES longword unsigned;                          /* COUNT OF VOLUMES MOUNTED
    LOGIN quadword unsigned;                            /* LOGIN TIME (100NS UNITS)
    OWNER longword unsigned;                            /* PID OF SUBPROCESS OWNER
    constant TERMLEN equals . prefix ACC$ tag K;        /* TERMINATION MESSAGE LENGTH
    constant TERMLEN equals . prefix ACC$ tag C;        /* TERMINATION MESSAGE LENGTH
                                                        /* AND ACCOUNTING RECORD FOR NON BATCH JOBS
                                                        /* END OF TERMINATION MESSAGE
    JOB_NAME character length 8;                        /* JOB NAME (BLANK FILLED)
    JOB_QUE character length 16;                        /* QUEUE NAME (.ASCIC)
    constant JOB_LEN equals . prefix ACC$ tag K;        /* LENGTH OF BATCH JOB ACCOUNTING RECORD
    constant JOB_LEN equals . prefix ACC$ tag C;        /* LENGTH OF BATCH JOB ACCOUNTING RECORD

end ACCDEF;

aggregate ACCDEF1 structure prefix ACC$;
    FILL_1 byte dimension 48 fill prefix ACCDEF tag $$;
    PAGCNT longword unsigned;                           /* SYMBIONT PAGE COUNT
    QIOCNT longword unsigned;                           /* SYMBIONT QIO COUNT
    GETCNT longword unsigned;                           /* SYMBIONT GET COUNT
    QUETIME quadword unsigned;                          /* TIME JOB WAS QUEUED
    PRT_NAME character length 8;                        /* NAME OF PRINT JOB
    PRT_QUE character length 12;                        /* NAME OF PRINT QUEUE
    constant PRT_LEN equals . prefix ACC$ tag K;        /* LENGTH OF PRINT ACCOUNTING RECORD
    constant PRT_LEN equals . prefix ACC$ tag C;        /* LENGTH OF PRINT ACCOUNTING RECORD

/*
/* DEFINE USER ACCOUNTING MESSAGE DATA AREA
/*
end ACCDEF1;

aggregate ACCDEF2 structure prefix ACC$;
    FILL_2 byte dimension 44 fill prefix ACCDEF tag $$;
    USER_DATA character length 132;                     /* ALLOW UP TO 132 BYTES OF USER DATA
    constant INS_LEN equals . prefix ACC$ tag K;        /* LENGTH OF INSERT MESSAGE
    constant INS_LEN equals . prefix ACC$ tag C;        /* LENGTH OF INSERT MESSAGE
/*
/* ASSIGN RECORD TYPE CODES FOR RECORDS IN THE ACCOUNT LOG FILE
/*

                                                        /* PROCESS ACCOUNTING RECORDS
    constant(
          PRCTRM                                        /* JOB TERMINATION
        , BATTRM                                        /* BATCH JOB TERMINATION
        , INTTRM                                        /* INTERACTIVE JOB TERMINATION
        , LOGTRM                                        /* LOGIN FAILURE PROCESS TERMINATION
        , IMGTRM                                        /* IMAGE TERMINATION
        , SUBTRM                                        /* SUBPROCESS TERMINATION
        , DETTRM                                        /* DETACHED PROCESS TERMINATION
        , NETTRM                                        /* NETWORK PROCESS TERMINATION
        ) equals 1 increment 1  prefix ACC tag $K;

                                                        /* MISC ACCOUNTING RECORDS
    constant(
          PRTJOB                                        /* PRINT JOB ACCOUNTING
        , INSMSG                                        /* INSERTED MESSAGE
        ) equals 16 increment 1  prefix ACC tag $K;

/*
/* DEFINE ACCOUNTING MANAGER MESSAGE CODES
/*


                                                        /*
    constant(
          INSMESG                                       /* INSERT MESSAGE INTO ACCOUNTING FILE
        , NEWFILE                                       /* CREATE A NEW ACCOUNT FILE
        , ENABACC                                       /* ENABLE ACCOUNTING
        , DISAACC                                       /* DISABLE ACCOUNTING
        , ENABSEL                                       /* ENABLE SELECTIVE ACCOUNTING
        , DISASEL                                       /* DISABLE SELECTIVE ACCOUNTING
        ) equals 1 increment 1  prefix ACC tag $K;

end ACCDEF2;

end_module $ACCDEF;
```

---

```
STA

modu

/*+
/*
/* E
/*
/*-

Cons




Cons

/*
/* F
/*

Cons






/* N

Cons









agg
```

```
module $ACEDEF;
/*+
/*
/* Access Control list Entry structure definitions
/*
/*-


aggregate ACEDEF structure prefix ACE$;
    SIZE byte unsigned;                         /* Size of the entry
    TYPE byte unsigned;                         /* Type of entry
    constant(
        KEYID                                   /* Key identifier entry
      , BIJNL                                   /* RMS BI journal name
      , AIJNL                                   /* RMS AI journal name
      , ATJNL                                   /* RMS AT journal name
      , AUDIT                                   /* Security audit journal entry
      , ALARM                                   /* Security alarm entry
      , INFO                                    /* General purpose information
      , JNLID                                   /* Journal-ID type
      , DIRDEF                                  /* Directory default protection
      ) equals 1 increment 1 tag C;
    FLAGS_OVERLAY union fill;
        FLAGS word unsigned;                    /* Type dependent & independent flags
        FLAGS_INFO structure fill;              /* Flags for INFO type ACE
            INFO_TYPE bitfield length 4;        /* INFO ACE subtype
            constant (
                CUST,                           /* Customer defined
                CSS                             /* CSS defined
                    ) equals 1 increment 1 tag C;
        end FLAGS_INFO;
        FLAGS_KEYID structure fill;             /* Flags for KEYID type
            RESERVED bitfield length 4;         /* Count of reserved longwords
        end FLAGS_KEYID;
        FLAGS_BITS0 structure fill;
            SUCCESS bitfield mask;              /* Audit or alarm upon success
            FAILURE bitfield mask;              /* Audit ot alarm upon failure
        end FLAGS_BITS0;
        FLAGS_BITS structure fill;              /* Type independent flags
            FILL_1 bitfield length 8 fill;
            "DEFAULT" bitfield mask;            /* Directory default entry
            PROTECTED bitfield mask;            /* Protected ACE
            HIDDEN bitfield mask;               /* Hidden ACE
            NOPROPAGATE bitfield mask;          /* No propagation between versions
        end FLAGS_BITS;
    end FLAGS_OVERLAY;

    ACE_FIELDS union fill;                      /* Start of ACE overlayed area
        KEY_AUD_TYPE structure fill;            /* KEY-ID and security audit types
            ACCESS structure longword unsigned; /* Access rights bitmask
                READ bitfield mask;             /* Allowed to read
                WRITE bitfield mask;            /* Allowed to write
                EXECUTE bitfield mask;          /* Allowed to execute
                DELETE bitfield mask;           /* Allowed to delete
                CONTROL bitfield mask;          /* All privileges of the owner
```

---

```
STA

end

end_
```

```
            end ACCESS;
            constant "LENGTH" equals . tag K;       /* Length of the overhead area
            constant "LENGTH" equals . tag C;       /* Length of the overhead area
            KEY_OVERLAY union fill;
                KEY longword unsigned;              /* Start of the key fields
                AUDITNAME character length 16;      /* Start of the security journal name
            end KEY_OVERLAY;
        end KEY_AUD_TYPE;

        INFO_TYPE structure fill;
            INFO_FLAGS longword unsigned;           /* INFO tyoe application flags
            INFO_START character length 1;          /* Start of the information
        end INFO_TYPE;

        RMSJNL_OVERLAY union fill;
            JNLID_TYPE structure fill;
                VOLNAM character length 12;         /* Volume name
                FID character length 6;             /* File-ID
                FILL_C3 word fill;
                ID_DATE quadword unsigned;          /* Time
            end JNLID_TYPE;
            RMSJNLNAM character length 16;          /* RMS journal name
        end RMSJNL_OVERLAY;

        DIRDEF_TYPE structure fill;
            SPARE1 longword unsigned;               /* For alignment
            SYS_PROT longword unsigned;             /* Default system protection
            OWN_PROT longword unsigned;             /* Default owner protection
            GRP_PROT longword unsigned;             /* Default group protection
            WOR_PROT longword unsigned;             /* Default world protection
        end DIRDEF_TYPE;
    end ACE_FIELDS;
end ACEDEF;

end_module $ACEDEF;
```

```
{+
{
{ Access Control List structure definitions
{
{-

module $ACLDEF;


aggregate ACLDEF structure prefix ACL$;
    FLINK longword unsigned;                            /* Forward link to next list in the queue
    BLINK longword unsigned;                            /* Back link to previous list in queue
    SIZE word unsigned;                                 /* Total size of the list
    TYPE byte unsigned;                                 /* Structure type code
    FILL_1 byte fill prefix ACLDEF tag $$;              /* Spare unused byte
    constant "LENGTH" equals . prefix ACL$ tag K;       /* Length of the overhead area
    constant "LENGTH" equals . prefix ACL$ tag C;       /* Length of the overhead area
    LIST longword unsigned;                             /* Start of the Access Control Entries

    constant (                                          /* Object types
          FILE                                          /* Files
        , DEVICE                                        /* MBX, MT, TT, etc.
        , JOBCTL_QUEUE                                  /* Job controller queue
        , COMMON_EF_CLUSTER                             /* Common event flag clusters
        , LOGICAL_NAME_TABLE                            /* Logical name tables
        , PROCESS                                       /* Process
        , GLOBAL_SECTION                                /* Global sections
        ) equals 1 increment 1 prefix ACL tag $C;

    constant (                                          /* Action codes
          ADDACLENT                                     /* Add an ACL entry
        , DELACLENT                                     /* Delete an ACL entry
        , MODACLENT                                     /* Modify an ACL entry
        , FNDACLENT                                     /* Locate an ACL entry
        , FNDACETYP                                     /* Locate specific ACE type
        , DELETEACL                                     /* Delete entire ACL
        , READACL                                       /* Read the ACL
        , ACLLENGTH                                     /* Get the ACL's length
        , READACE                                       /* Read a single ACE
        , RLOCK_ACL                                     /* Read lock on ACL
        , WLOCK_ACL                                     /* Write lock on ACL
        , UNLOCK_ACL                                    /* Release exclusive lock
        ) equals 1 increment 1 prefix ACL tag $C;

    constant ADDACLENT  equals 255 prefix ACL tag $S;   /* Add an ACL entry
    constant DELACLENT  equals 255 prefix ACL tag $S;   /* Delete an ACL entry
    constant MODACLENT  equals 255 prefix ACL tag $S;   /* Modify an ACL entry
    constant FNDACLENT  equals 255 prefix ACL tag $S;   /* Locate an ACL entry
    constant FNDACETYP  equals 255 prefix ACL tag $S;   /* Locate specific ACE type
    constant DELETEACL  equals 255 prefix ACL tag $S;   /* Delete entire ACL
    constant READACL    equals 512 prefix ACL tag $S;   /* Read the ACL
    constant ACLLENGTH  equals   4 prefix ACL tag $S;   /* Get the ACL's length
    constant READACE    equals 255 prefix ACL tag $S;   /* Read a single ACE
    constant RLOCK_ACL  equals   4 prefix ACL tag $S;   /* Read lock on ACL
    constant WLOCK_ACL  equals   4 prefix ACL tag $S;   /* Write lock on ACL
    constant UNLOCK_ACL equals   4 prefix ACL tag $S;   /* Remove lock on ACL
```

---

```
STA

modu
/*-
/*
/* [
/*+
agg




end

end_
```

```
end ACLDEF;

end_module $ACLDEF;
```

```
module $ACRDEF;
/*+
/* ACRDEF - ACCOUNTING RECORD DEFINITIONS
/*
/*********************************************************************
/* NOTE:  IF ANY FIELDS CHANGE, A NEW VERSION NUMBER MUST BE ADDED AND *
/*          ''ACR$K_CURVER'' EQUATED TO IT.                                  *
/*********************************************************************
/*
/*-


                                                            /* RECORD/PACKET VERSIONS (ACR$V_VERSION)
constant(
      VERSION2                                              /* VMS VERSION 2 ACCOUNTING FORMAT
    , VERSION3T                                             /* VMS VERSION 3 FIELD TEST
    , VERSION3                                              /* VMS VERSION 3 ACCOUNTING FORMAT
    ) equals 0 increment 1  prefix ACR tag $K;
constant(
      CURVER                                                /* CURRENT FORMAT VERSION NUMBER
    ) equals ACR$K_VERSION3 increment 0  prefix ACR tag $K;


aggregate ACRDEF structure prefix ACR$;
    TYPE_OVERLAY union fill;
        TYPE word unsigned;                                 /* RECORD/PACKET TYPE

        TYPE_BITS structure fill;
            PACKET bitfield mask;                           /* RECORD(0)/PACKET(1)
            TYPE bitfield mask length 7;                    /* RECORD/PACKET TYPE
            SUBTYPE bitfield mask length 4;                 /* RECORD/PACKET SUBTYPE
            VERSION bitfield mask length 3;                 /* RECORD/PACKET VERSION NUMBER
            CUSTOMER bitfield mask;                         /* DIGITAL(0)/CUSTOMER(1)
        end TYPE_BITS;

                                                            /* RECORD TYPE (ACR$V_TYPE) CONSTANTS
        constant(
              PRCDEL                                        /* PROCESS DELETE
            , PRCPUR                                        /* PROCESS PURGE
            , IMGDEL                                        /* IMAGE DELETE
            , IMGPUR                                        /* IMAGE PURGE
            , SYSINIT                                       /* SYSTEM INITIALIZATION
            , SETTIME                                       /* SET SYSTEM TIME
            , LOGFAIL                                       /* LOGIN VALIDATION FAILURE
            , PRINT                                         /* PRINT JOB
            , USER                                          /* USER SUPPLIED DATA
            , ENABLE                                        /* ACC. MANG. FUNCTION ENABLE
            , DISABLE                                       /* ACC. MANG. FUNCTION DISABLE
            , ALTACM                                        /* DECLARE ALTERNATE ACC. MANG.
            , FILE_FL                                       /* ACCOUNTING FILE - FORWARD LINK
            , FILE_BL                                       /* ACCOUNTING FILE - BACKWARD LINK
            ) equals 1 increment 1  prefix ACR tag $K;

                                                            /* RECORD SUBTYPE (ACR$V_SUBTYPE) CONSTANTS
        constant(
              INTERACTIVE                                   /* INTERACTIVE PROCESS
```

```
STA

modu
/**
/*
/*
/*
/*
/*
/*
/*
/*-

agg









/*



/*
/*
/*

end

agg
```

```
            , SUBPROCESS                                         /* SUBPROCESS
            , DETACHED                                           /* DETACHED PROCESS
            , BATCH                                              /* BATCH PROCESS
            , NETWORK                                            /* NETWORK PROCESS
            ) equals 1 increment 1  prefix ACR tag $K;

                                                                 /* PACKET TYPE (ACR$V_TYPE) CONSTANTS
        constant(
              ID                                                 /* IDENTIFICATION PACKET
            , RESOURCE                                           /* RESOURCE USAGE PACKET
            , IMAGENAME                                          /* IMAGENAME PACKET
            , FILENAME                                           /* FILENAME PACKET
            , USER_DATA                                          /* USER DATA PACKET
            ) equals 1 increment 1  prefix ACR tag $K;

    end TYPE_OVERLAY;
    "LENGTH" word unsigned;                                      /* RECORD OR PACKET LENGTH
/*
/* RECORD HEADER
/*
end ACRDEF;

aggregate ACRDEF1 structure prefix ACR$;
    FILL_2 byte dimension 4 fill prefix ACRDEF tag $$;
    SYSTIME quadword unsigned;                                   /* EVENT SYSTEM TIME
    constant HDRLEN equals . prefix ACR$ tag K;                  /* RECORD HEADER LENGTH
    constant HDRLEN equals . prefix ACR$ tag C;                  /* RECORD HEADER LENGTH

/*
/* IDENTIFICATION PACKET
/*
end ACRDEF1;

aggregate ACRDEF2 structure prefix ACR$;
    FILL_3 byte dimension 4 fill prefix ACRDEF tag $$;
    PID longword unsigned;                                       /* PROCESS ID
    OWNER longword unsigned;                                     /* OWNER PROCES ID
    UIC_OVERLAY union fill;
        UIC longword unsigned;                                   /* PROCESS UIC
        UIC_FIELDS structure fill;
            MEM word unsigned;                                   /* MEMBER UIC
            GRP word unsigned;                                   /* GROUP UIC
        end UIC_FIELDS;
    end UIC_OVERLAY;
    PRIV quadword unsigned;                                      /* PROCESS PRIV
    PRI byte unsigned;                                           /* PROCESS PRIORITY
    FILL_1 byte fill prefix ACRDEF tag $$;                       /* SPARE
    USERNAME word unsigned;                                      /* USERNAME OFFSET
    ACCOUNT word unsigned;                                       /* ACCOUNT NAME OFFSET
    NODENAME word unsigned;                                      /* NODE NAME OFFSET
    TERMINAL word unsigned;                                      /* TERMINAL NAME OFFSET
    JOBNAME word unsigned;                                       /* JOB NAME OFFSET
    JOBID longword unsigned;                                     /* JOB ID
    QUEUE word unsigned;                                         /* QUEUE NAME OFFSET
    NODEADDR word unsigned;                                      /* REMOTE NODE ADDRESS
    REMOTEID word unsigned;                                      /* REMOTE ID OFFSET
    constant IDVAR equals . prefix ACR$ tag K;          /* BEGINNING OF VARIABLE STORAGE AREA
    constant IDVAR equals . prefix ACR$ tag C;          /* BEGINNING OF VARIABLE STORAGE AREA
/*
/* RESOURCE PACKET
/*
end ACRDEF2;

aggregate ACRDEF3 structure prefix ACR$;
    FILL_4 byte dimension 4 fill prefix ACRDEF tag $$;
    LOGIN quadword unsigned;                            /* PROCESS/IMAGE START TIME
    STATUS longword unsigned;                           /* PROCESS/IMAGE FINAL STATUS
    IMGCNT longword unsigned;                           /* IMAGE EXECUTION COUNT/SEQUENCE NUMBER
    CPUTIME longword unsigned;                          /* PROCESS/IMAGE CPU TIME
    FAULTS longword unsigned;                           /* PROCESS/IMAGE PAGE FAULT COUNT
    FAULTIO longword unsigned;                          /* PROCESS/IMAGE PAGE FAULT I/O COUNT
    WSPEAK longword unsigned;                           /* PROCESS/IMAGE WORKING SET PEAK
    PAGEFL longword unsigned;                           /* PROCESS/IMAGE PEAK PAGE FILE USAGE
    DIOCNT longword unsigned;                           /* PROCESS/IMAGE DIRECT I/O COUNT
    BIOCNT longword unsigned;                           /* PROCESS/IMAGE BUFFERED I/O COUNT
    VOLUMES longword unsigned;                          /* PROCESS/IMAGE VOLUME MOUNT COUNT

/*
/* IMAGENAME PACKET
/*
end ACRDEF3;

aggregate ACRDEF4 structure prefix ACR$;
    FILL_5 byte dimension 4 fill prefix ACRDEF tag $$;
    IMAGENAME character length 256;                     /* IMAGENAME
/*
/* PRINT RESOURCE PACKET
/*
end ACRDEF4;

aggregate ACRDEF5 structure prefix ACR$;
    FILL_6 byte dimension 4 fill prefix ACRDEF tag $$;
    PRINTSTS longword unsigned;                         /* JOB STATUS
    QUETIME quadword unsigned;                          /* TIME JOB WAS QUEUED
    BEGTIME quadword unsigned;                          /* TIME JOB WAS BEGUN
    SYMCPUTIM longword unsigned;                        /* SYMBIONT CPU TIME
    PAGECNT longword unsigned;                          /* TOTAL PAGES PRINTED
    QIOCNT longword unsigned;                           /* TOTAL QIOS ISSUED
    GETCNT longword unsigned;                           /* TOTAL GETS ISSUED
/*
/* FILENAME PACKET
/*
end ACRDEF5;

aggregate ACRDEF6 structure prefix ACR$;
    FILL_7 byte dimension 4 fill prefix ACRDEF tag $$;
    FILENAME character length 256;                      /* FILENAME
/*
/* USER DATA PACKET
/*
end ACRDEF6;
```

```
aggregate ACRDEF7 structure prefix ACR$;
    FILL_8 byte dimension 4 fill prefix ACRDEF tag $$;
    USER_DATA character length 256;                  /* USER DATA
end ACRDEF7;

end_module $ACRDEF;
```

---

```
/*
/*
/*
/*

/*
/*
/*

end

agg
```

```
module $ARMDEF;

/*+
/* Access Rights Mask longword definitions
/*-

aggregate ARMDEF structure prefix ARM$;
        READ bitfield mask;             /* Read access
        WRITE bitfield mask;            /* Write access
        EXECUTE bitfield mask;          /* Execute access
        DELETE bitfield mask;           /* Delete access
        CONTROL bitfield mask;          /* Control access (modify attributes)
        FILL bitfield length 32-^;
end ARMDEF;

end_module $ARMDEF;
```

```
module $ATRDEF;
/* ATTRIBUTE LIST DESCRIPTION. THE ATTRIBUTE CONTROL LIST IS USED TO READ AND
/* WRITE FILE ATTRIBUTES. IT CONSISTS OF CONCATENATED ATTRIBUTE CONTROL BLOCKS
/* TERMINATED BY A SINGLE ZERO LONGWORD.
/*


aggregate ATRDEF structure prefix ATR$;
    SIZE word unsigned;                                 /* SIZE OF ATTRIBUTE IN BYTES
    TYPE word unsigned;                                 /* ATTRIBUTE TYPE CODE
    ADDR longword unsigned;                             /* ADDRESS OF ATTRIBUTE TEXT

                                                        /* ATTRIBUTE CODES
    constant(
         UCHAR                                          /* 4 BYTE USER FILE CHARACTERISTICS
        , RECATTR                                       /* 32 BYTES RECORD ATTRIBUTES
        , FILNAM                                        /* 6 BYTE RAD-50 FILE NAME
        , FILTYP                                        /* 2 BYTE RAD-50 FILE TYPE
        , FILVER                                        /* 2 BYTE BINARY FILE VERSION
        , EXPDAT                                        /* 7 BYTE ASCII EXPIRATION DATE
        , STATBLK                                       /* 32 BYTE STATISTICS BLOCK
        , HEADER                                        /* 512 BYTE FILE HEADER
        , BLOCKSIZE                                     /* MAGTAPE BLOCK SIZE
        , USERLABEL                                     /* USER FILE LABEL
        , ASCDATES                                      /* REVISION COUNT THRU EXP DATE IN ASCII
        , ALCONTROL                                     /* COMPATIBILITY MODE ALLOCATION DATA
        , ENDLBLAST                                     /* END OF MAGTAPE LABEL PROCESSING AND SUPPLY AST CONTROL BLOCK
        , ASCNAME                                       /* FILE NAME, TYPE & VERSION IN ASCII
        , CREDATE                                       /* 64 BIT CREATION DATE
        , REVDATE                                       /* 64 BIT REVISION DATE
        , EXPDATE                                       /* 64 BIT EXPIRATION DATE
        , BAKDATE                                       /* 64 BIT BACKUP DATE
        , UIC                                           /* 4 BYTE FILE OWNER UIC
        , FPRO                                          /* 2 BYTE FILE PROTECTION
        , RPRO                                          /* 2 BYTE RECORD PROTECTION
        , ACLEVEL                                       /* 1 BYTE FILE ACCESS LEVEL
        , SEMASK                                        /* FILE SECURITY MASK AND LIMIT
        , UIC_RO                                        /* READ ONLY UIC
        , DIRSEQ                                        /* DIRECTORY UPDATE SEQUENCE COUNT
        , BACKLINK                                      /* FILE BACK LINK POINTER
        , JOURNAL                                       /* JOURNAL CONTROL FLAGS
        , HDR1_ACC                                      /* ANSI TAPE HEADER 1 ACCESSIBILITY
                                                        /*  CHARACTER
        , ADDACLENT                                     /* ADD AN ACCESS CONTROL ENTRY
        , DELACLENT                                     /* REMOVE AN ACCESS CONTROL ENTRY
        , MODACLENT                                     /* MODIFY AN ACL ENTRY
        , FNDACLENT                                     /* LOCATE AN ACL ENTRY
        , FNDACLTYP                                     /* FIND A SPECIFIC TYPE OF ACE
        , DELETEACL                                     /* DELETE THE ENTIRE ACL
        , READACL                                       /* READ THE ENTIRE ACL
        , ACLLENGTH                                     /* RETURN THE LENGTH OF THE ACL
        , READACE                                       /* READ A SINGLE ACE
        , RESERVED                                      /* MODIFY RESERVED AREA
        , HIGHWATER                                     /* HIGHWATER MARK (USER READ ONLY)
        , DUMMY_0                                       /* *** AVAILABLE CODE
```

---

```
STA



/*
/*
/*




/*
/*
/*




/*
/*
/*




/*
/*
/*




/*
/*
/*






end

end

mod
```

```
        , PRIVS_USED                                        /* PRIVILEGES USED TO GAIN ACCESS
        , MATCHING_ACE                                      /* ACE USED TO GAIN ACCESS (IF ANY)
        , ACCESS_MODE                                       /* ACCESS MODE FOR FOLLOWING ATTRIBUTE DESCRIPTORS
        , FILE_SPEC                                         /* CONVERT FID TO FILE-SPEC
        , CLASS_MASK                                        /* Non-discretionary classification mask
        , BUFFER_OFFSET                                     /* For magnetic tape only length of buffer offset of block in file

                                                            /* All new attributes should be
                                                            /*  added here before MAX_PLUS1


        , MAX_PLUS1                                         /* Maximum code plus one
        ) equals 3 increment 1 prefix ATR tag $C;

    constant MAX_CODE   equals (ATR$C_MAX_PLUS1 - 1)  prefix ATR tag $C;
    constant FNDACETYP equals ATR$C_FNDACLTYP prefix ATR tag $C;

                                                            /* ATTRIBUTE MAXIMUM LENGTHS
    constant UCHAR       equals 4  prefix ATR tag $S;       /* 4 BYTE USER FILE CHARACTERISTICS
    constant RECATTR     equals 32  prefix ATR tag $S;      /* 32 BYTES RECORD ATTRIBUTES
    constant FILNAM      equals 6  prefix ATR tag $S;       /* 6 BYTE RAD-50 FILE NAME
    constant FILTYP      equals 2  prefix ATR tag $S;       /* 2 BYTE RAD-50 FILE TYPE
    constant FILVER      equals 2  prefix ATR tag $S;       /* 2 BYTE BINARY FILE VERSION
    constant EXPDAT      equals 7  prefix ATR tag $S;       /* 7 BYTE ASCII EXPIRATION DATE
    constant STATBLK     equals 32  prefix ATR tag $S;      /* 32 BYTE STATISTICS BLOCK
    constant HEADER      equals 512  prefix ATR tag $S;     /* 512 BYTE FILE HEADER
    constant BLOCKSIZE   equals 2  prefix ATR tag $S;       /* MAGTAPE BLOCK SIZE
    constant USERLABEL   equals 80  prefix ATR tag $S;      /* USER FILE LABEL
    constant ASCDATES    equals 35  prefix ATR tag $S;      /* REVISION COUNT THRU EXP DATE IN ASCII
    constant ALCONTROL   equals 14  prefix ATR tag $S;      /* COMPATIBILITY MODE ALLOCATION DATA
    constant ENDLBLAST   equals 4  prefix ATR tag $S;       /* END OF MAGTAPE LABEL PROCESSING AND SUPPLY AST CONTROL BLOCK
    constant ASCNAME     equals 86  prefix ATR tag $S;      /* FILE NAME, TYPE & VERSION IN ASCII
    constant CREDATE     equals 8  prefix ATR tag $S;       /* 64 BIT CREATION DATE
    constant REVDATE     equals 8  prefix ATR tag $S;       /* 64 BIT REVISION DATE
    constant EXPDATE     equals 8  prefix ATR tag $S;       /* 64 BIT EXPIRATION DATE
    constant BAKDATE     equals 8  prefix ATR tag $S;       /* 64 BIT BACKUP DATE
    constant UIC         equals 4  prefix ATR tag $S;       /* 4 BYTE FILE OWNER UIC
    constant FPRO        equals 2  prefix ATR tag $S;       /* 2 BYTE FILE PROTECTION
    constant RPRO        equals 2  prefix ATR tag $S;       /* 2 BYTE RECORD PROTECTION
    constant ACLEVEL     equals 1  prefix ATR tag $S;       /* 1 BYTE FILE ACCESS LEVEL
    constant SEMASK      equals 8  prefix ATR tag $S;       /* FILE SECURITY MASK AND LIMIT
    constant UIC_RO      equals 4  prefix ATR tag $S;       /* READ ONLY UIC
    constant DIRSEQ      equals 2  prefix ATR tag $S;       /* DIRECTORY UPDATE SEQUENCE COUNT
    constant BACKLINK    equals 6  prefix ATR tag $S;       /* FILE BACK LINK POINTER
    constant JOURNAL     equals 2  prefix ATR tag $S;       /* JOURNAL CONTROL FLAGS
    constant HDR1_ACC    equals 1  prefix ATR tag $S;       /* ANSI TAPE HEADER 1 ACCESSIBILITY
                                                            /*  CHARACTER
    constant ADDACLENT   equals 255  prefix ATR tag $S;     /* ADD AN ACCESS CONTROL ENTRY
    constant DELACLENT   equals 255  prefix ATR tag $S;     /* REMOVE AN ACCESS CONTROL ENTRY
    constant MODACLENT   equals 255  prefix ATR tag $S;     /* MODIFY AN ACL ENTRY
    constant FNDACLENT   equals 255  prefix ATR tag $S;     /* LOCATE AN ACL ENTRY
    constant FNDACLTYP   equals 255  prefix ATR tag $S;     /* FIND A SPECIFIC TYPE OF ACE
    constant FNDACETYP   equals 255  prefix ATR tag $S;     /* FIND A SPECIFIC TYPE OF ACE
    constant DELETEACL   equals 255  prefix ATR tag $S;     /* DELETE THE ENTIRE ACL
    constant READACL     equals 512  prefix ATR tag $S;     /* READ THE ENTIRE ACL
    constant ACLLENGTH   equals 4  prefix ATR tag $S;       /* RETURN THE LENGTH OF THE ACL
    constant READACE equals 255 prefix ATR tag $S;          /* READ A SINGLE ACE
    constant RESERVED equals 380 prefix ATR tag $S;      /* MODIFY RESERVED AREA
    constant HIGHWATER equals 4 prefix ATR tag $S;       /* FILE HIGH WATER MARK (USER READ ONLY)
    constant DUMMY_0 equals 4 prefix ATR tag $S;         /* *** AVAILABLE CODE
    constant PRIVS_USED equals 4 prefix ATR tag $S;      /* PRIVS USED TO GAIN ACCESS
    constant MATCHING_ACE equals 255 prefix ATR tag $S; /* ACE USED TO GAIN ACCESS
    constant ACCESS_MODE equals 1 prefix ATR tag $S;     /* ACCESS MODE FOR FOLLOWING ATTRIBUTE DESCRIPTORS
    constant FILE_SPEC equals 512 prefix ATR tag $S;     /* CONVERT FID TO FILE-SPEC
    constant CLASS_MASK equals 20 prefix ATR tag $S;     /* Non-discretionary classification mask
    constant BUFFER_OFFSET equals 2  prefix ATR tag $S; /* Buffer offset length field
end ATRDEF;

end_module $ATRDEF;
```

```
module $BRKDEF;

/*+
/*
/* Breakthru system service input definitions.
/*
/*-

Constant (
            DEVICE,                     /* device name
            USERNAME,                   /* user name
            ALLUSERS,                   /* all users
            ALLTERMS                    /* all logged in users
        ) Equals 1 Increment 1 PREFIX BRK$ TAG "C";

Constant MAXSENDTYPE Equals BRK$C_ALLTERMS PREFIX BRK$ TAG "C";

/*
/* Requestor ID's, DEC use only (0-31)
/*

Constant (
        GENERAL,                        /* GENERAL (OR UNSPECIFIED)
        PHONE,                          /* PHONE
        MAIL,                           /* MAIL
        QUEUE,                          /* QUEUE MANAGER
        SHUTDOWN,                       /* SYSTEM SHUTDOWN
        URGENT,                         /* URGENT MESSAGE
        DCL,                            /* DCL (control T)
        OPCOM                           /* OPERATOR MESSAGE
         ) Equals 0 Increment 1 PREFIX BRK$ TAG "C";

/* Note that only first 16 are really stored by TTDRIVER now

Constant (
        USER1,                          /* reserved to customer
        USER2,                          /* reserved to customer
        USER3,                          /* reserved to customer
        USER4,                          /* reserved to customer
        USER5,                          /* reserved to customer
        USER6,                          /* reserved to customer
        USER7,                          /* reserved to customer
        USER8,                          /* reserved to customer
        USER9,                          /* reserved to customer
        USER10,                         /* reserved to customer
        USER11,                         /* reserved to customer
        USER12,                         /* reserved to customer
        USER13,                         /* reserved to customer
        USER14,                         /* reserved to customer
        USER15,                         /* reserved to customer
        USER16                          /* reserved to customer
         ) Equals 32 Increment 1 PREFIX BRK$ TAG "C";

aggregate FLAGS_INPUT structure prefix BRK$;    /* mimics $BRDCSTDEF
```

---

```
STA

mod

/*+
/*
/*
/*
/*
/*-

agg



end

end
```

```
        ERASE_LINES     bitfield length 8;      /* number of lines to erase
        SCREEN          bitfield mask;          /* Do screen formatted write
        BOTTOM          bitfield mask;          /* "screen" message at bottom
        NOREFRESH       bitfield mask;          /* Refresh an interrupted read
        CLUSTER         bitfield mask;          /* broadcast to cluster

end FLAGS_INPUT;

end_module $BRKDEF;
```

---

```
STA

mod
/*+
/*
/*-


agg



/*
/*
/*




end

end
```

```
module $CHFDEF;
/*
/* CONDITION HANDLING ARGUMENT LIST OFFSETS
/* THERE ARE THREE CONDITIONAL HANDLING STRUCTURES: THE PRIMARY ARGUMENT
/* LIST, AND THE SIGNAL AND MECHANISM ARRAYS. ALL ARE IDENTIFIED BY THE SAME
/* BLOCK PREFIX.
/*


aggregate CHFDEF structure prefix CHF$;
    FILL_1 longword fill prefix CHFDEF tag $$;             /*PRIMARY ARGUMENT COUNT
    SIGARGLST longword unsigned;                           /*ADDRESS OF SIGNAL ARGUMENTS
    MCHARGLST longword unsigned;                           /*ADDRESS OF MECHANISM ARGUMENTS

end CHFDEF;

aggregate CHFDEF1 structure prefix CHF$;
    SIG_ARGS longword unsigned;                            /*NUMBER OF SIGNAL ARGUMENTS
    SIG_NAME longword unsigned;                            /*SIGNAL NAME
    SIG_ARG1 longword unsigned;                            /*FIRST SIGNAL SPECIFIC ARGUMENT

end CHFDEF1;

aggregate CHFDEF2 structure prefix CHF$;
    MCH_ARGS longword unsigned;                            /*NUMBER OF MECHANISM ARGUMENTS
    MCH_FRAME longword unsigned;                           /*ESTABLISHER FRAME ADDRESS
    MCH_DEPTH longword unsigned;                           /*FRAME DEPTH OF ESTABLISHER
    MCH_SAVR0 longword unsigned;                           /*SAVED REGISTER R0
    MCH_SAVR1 longword unsigned;                           /*SAVED REGISTER R1

end CHFDEF2;

end_module $CHFDEF;
```

```
module $CHKPNTDEF;
/*-
/*
/* Define flags for calls to create checkpointable processes
/*+
aggregate CHKPNTDEF structure prefix CHKPNT$;

        AUTO_RESTART    bitfield mask;          /* auto-restart detached process
        DEBUG           bitfield mask;          /* merge in DEBUG
        ASYNC           bitfield mask;          /* asynchronous checkpoint process

end CHKPNTDEF;

end_module $CHKPNTDEF;
```

```
module $CHPDEF;

/*+
/*
/* Item definitions for the $CHKPRO (check protection) service.
/*
/*-

constant (
        END,                            /* list end
        ACCESS,                         /* intended access
        FLAGS,                          /* read / write flags
        PRIV,                           /* privilege mask
        ACMODE,                         /* accessor access mode
        ACCLASS,                        /* accessor security classification
        RIGHTS,                         /* rights list
        ADDRIGHTS,                      /* additional rights list segments
        MODE,                           /* simple access mode
        MODES,                          /* per action access mode
        MINCLASS,                       /* minimum security classification
        MAXCLASS,                       /* maximum security classification
        OWNER,                          /* object owner UIC
        PROT,                           /* protection mask
        ACL,                            /* access control list segment
        AUDITNAME,                      /* security audit name
        ALARMNAME,                      /* security alarm name
        MATCHEDACE,                     /* address of matching ACE
        PRIVUSED,                       /* mask of privileges used

        MAX_CODE                        /* highest CHP$ item code
        ) equals 0 increment 1 prefix CHP tag $;

aggregate PRIVS_USED_BITS structure prefix CHP$;
    SYSPRV bitfield mask;               /* access via SYSPRV privilege
    BYPASS bitfield mask;               /* access via BYPASS privilege
    UPGRADE bitfield mask;              /* access via UPGRADE privilege
    DOWNGRADE bitfield mask;            /* access via DOWNGRADE privilege
    GRPPRV bitfield mask;               /* access via GRPPRV privilege
    READALL bitfield mask;              /* access via READALL privilege
end PRIVS_USED_BITS;

aggregate FLAG_BITS structure prefix CHP$;
    READ bitfield mask;                 /* non-discretionary read access
    WRITE bitfield mask;                /* non-discretionary write access
    USEREADALL bitfield mask;           /* READALL privilege applies
end FLAG_BITS;

end_module $CHPDEF;
```

---

```
STA

/*
/*

con
con
con
con
con
con
con
con
con
con

con
con
con
con
con
con
con
con
con
con
con
con
con
con
con
con
con
con
con
con
con
con
con
con
con
con
con
con
/*
/*
/*
con
con
con
con
/*
/*
/*
con
con
con
```

```
module $CLIDEF;
/*++
/*
/* $CLIDEF --- COMMAND LAGUAGE INTERFACE DEFINITIONS
/*
/* THIS MODULE DEFINES THE OFFSET VALUES FOR THE DATA STRUCTURES
/* USED TO COMMUNICATED COMMAND INFORMATION BETWEEN THE CLI AND
/* THE UTILITY THAT WAS REQUESTED TO PERFORM THE OPERATION.
/*
/*--


aggregate CLIDEF structure prefix CLI$;
    INIARGCNT longword unsigned;                           /* INITIAL ARG COUNT
    PROGXFER address;                                      /* PROGRAM TRANSFER VECTOR ADDRESS
    UTILSERV address;                                      /* ADDRESS OF CLI UTILITY SERVICE
    IMGHDADR address;                                      /* ADDRESS OF IMAGE HEADER
    IMGFILED address;                                      /* ADDRESS OF IMAGE FILE DATA
    LINKFLAG longword unsigned;                            /* LINK FLAG LONGWORD
    CLIFLAG_OVERLAY union fill;
        CLIFLAG longword unsigned;                         /* CLI FLAG LONGWORD
        CLIFLAG_BITS structure fill;
            DEBUG bitfield mask;                           /* DEBUG QUALIFIER SEEN
            DBGTRU bitfield mask;                          /* DEBUG WAS TRUE
            VERIFY bitfield mask;                          /* VERIFY IS ON
            BATCH bitfield mask;                           /* THIS PROCESS IS A BATCH JOB
            INDIRECT bitfield mask;                        /* INDIRECT LEVEL NOT EQUAL TO ZERO
            VFYINP bitfield mask;                          /* UTILITY SHOULD VERFIY INPUT
            FILL_1 bitfield length 2 fill prefix CLIDEF tag $$;/* ROUND UP TO NEXT BYTE
            TRMVRBLV bitfield mask length 2;               /* TERMINAL VERBOSITY LEVEL
            FILL_2 bitfield length 6 fill prefix CLIDEF tag $$;/* ROUND UP TO NEXT BYTE
            DBGEXCP bitfield mask;                         /* DEBUGGER CALLED VIA "DEBUG" EXCEPTION
        end CLIFLAG_BITS;
/*        THE FOLLOWING ARGUMENTS ARE OPTIONAL DEPENDING ON INIARGCNT
    end CLIFLAG_OVERLAY;
    ARGLIST longword unsigned;                             /* ADDRESS OF APPLICATION ARGUMENT LIST
                                                           /* (USED FOR MERGED IMAGE ARG. PASSING)

/*
/*        DEFINE CLI COMMAND INTERFACE REQUEST BLOCK
/*


end CLIDEF;

aggregate CLIDEF1 structure prefix CLI$;
    FILL_3_OVERLAY union fill;
        FILL_3 longword fill prefix CLIDEF tag $$;       /* DUMMY ARG
        FILL_3_FIELDS structure fill;
            RQTYPE_OVERLAY union fill;
                RQTYPE byte unsigned;                    /* TYPE OF REQUEST BEING MADE
                RQTYPE_BITS structure fill;
                    SUBTYP bitfield length 4;            /* SUB TYPE VIELD
                    PRITYP bitfield length 4;            /* PRIMARY TYPYE
                end RQTYPE_BITS;
            end RQTYPE_OVERLAY;
            SERVCOD_OVERLAY union fill;
                SERVCOD word unsigned;                  /* CLI SERVICE CODE
                SERVCOD_FIELDS structure fill;
                    RQINDX_OVERLAY union fill;
                        RQINDX byte unsigned;           /* OFFSET FOR VALUE KEYWORD INDEX
                        BITNUM byte unsigned;           /* BIT TO SET IF REQUEST IS SUCESSFUL
                    end RQINDX_OVERLAY;
                    RQFLGS_OVERLAY union fill;
                        RQFLGS byte unsigned;           /* INPUT FLAGS CONCERNING REQUEST
                        RQFLGS_BITS0 structure fill;
                            PARMREQ bitfield mask;      /* PARAMATER IS REQUIRED
                            ABSADR bitfield mask;       /* ALL ADDRESS ARE ABSOLUTE
                            EXPNAM bitfield mask;       /* RETURN EXPLICIT NAMES ONLY
                        end RQFLGS_BITS0;

                        RQFLGS_BITS1 structure fill;
                            LASTVAL bitfield mask;      /* LAST VALUE ALLOWED
                            DUMMY bitfield mask;        /* *** SAVE PLACE FOR ABS ADR ***
                        end RQFLGS_BITS1;

                    end RQFLGS_OVERLAY;
                end SERVCOD_FIELDS;
            end SERVCOD_OVERLAY;
            RQSTAT_OVERLAY union fill;
                RQSTAT byte unsigned;                   /* OUTPUT STATUS FLAGS

                RQSTAT_BITS0 structure fill;
                    PARMPRS bitfield mask;              /* PARAMETER IS PRESENT
                    CONCATINP bitfield mask;            /* INPUT CONCATONATION EXISTS
                    MOREINP bitfield mask;              /* ANOTHER SET OF INPUT PARAMETERS
                    PARMDEF bitfield mask;              /* PARAMETER WAS DEFAULTED PRESENT
                end RQSTAT_BITS0;
                RQSTAT_BITS1 structure fill;
                    MOREVALS bitfield mask;             /* UNPROCESSED VALUES REMAIN
                    KEYVALU bitfield mask;              /* SUBSEQUENT TOKEN IS VALUE FOR KEY
                end RQSTAT_BITS1;

            end RQSTAT_OVERLAY;
        end FILL_3_FIELDS;
    end FILL_3_OVERLAY;
    ERRACT address;                                     /* ADDRESS OF ERROR ACTION ROUTINE
    RQDESC_OVERLAY union fill;
        RQDESC quadword unsigned;                       /* QUADWORD REQUEST DESCRIPTOR
        RQDESC_FIELDS structure fill;
            RQSIZE word unsigned;                       /* RESULTANT SIZE OF PARAMETER
            FILL_4 word fill prefix CLIDEF tag $$;      /* SPARE WORD IN DESCRIPTOR
            RQVALU_OVERLAY union fill;
                RQVALU longword unsigned;               /* ACTUAL VALUE IN VALUE CONVERSTION
                RQADDR address;                         /* ADDRESS OF RESULTANT PARAMETER
            end RQVALU_OVERLAY;
        end RQDESC_FIELDS;
    end RQDESC_OVERLAY;
    PRSACT address;                                     /* PARAMETER PRESENT ACTION ROUTINE
    ABSACT address;                                     /* PARAMETER ABSENT ACTION ROUTINE
    QUALST address;                                     /* ADDRESS OF QUALIFIERS
    constant REQDESC equals . prefix CLI$ tag K;        /* SIZE OF THE STRUCTURE
    constant REQDESC equals . prefix CLI$ tag C;          /* SIZE OF THE STRUCTURE

/*
/* Define the Descriptor Used by Symbol and Logical-name Callbacks
/*
end CLIDEF1;

aggregate CLIDEF2 structure prefix CLI$;
    FILL_8 byte dimension 4 fill prefix CLIDEF tag $$;
    NAMDESC quadword unsigned;                            /* Logical name or symbol name
    VALDESC quadword unsigned;                            /* Equivalence name or symbol value
    TABDESC quadword unsigned;                            /* Name of logical name table
    ITMLST longword unsigned;                             /* Address of item list
    ATTR longword unsigned;                               /* Address of attribute longword

/*
/* Define the Descriptor Locations Used for the Old/New
/* Out-of-Band Character(s) Masks
/*
end CLIDEF2;

aggregate CLIDEF3 structure prefix CLI$;
    FILL_9 byte dimension 4 fill prefix CLIDEF tag $$;
    NEW_MASK longword unsigned;                           /* Enable/disable mask
    OLD_MASK longword unsigned;                           /* Previous enabled values mask

/*
/* Define the descriptor used for ATTACH callback
/*

end CLIDEF3;

aggregate CLIDEF4 structure prefix CLI$;
    FILL_10 byte dimension 4 fill prefix CLIDEF tag $$;
    PID longword unsigned;                                /* PID of "destination" process

/*
/* Define the descriptor used for SPAWN callback
/*
end CLIDEF4;

aggregate CLIDEF5 structure prefix CLI$;
    FILL_11 byte dimension 4 fill prefix CLIDEF tag $$;
    FLAGS_OVERLAY union fill;
        FLAGS byte unsigned;                              /* Flags byte
        FLAGS_BITS structure fill;
            NOWAIT bitfield mask;                         /* Do not wait for subprocess completion
            NOCLISYM bitfield mask;                       /* Do not copy CLI symbols to subprocess
            NOLOGNAM bitfield mask;                       /* Do not copy logical names to subprocess
            NOKEYPAD bitfield mask;                       /* Do not copy keypad state to subprocess
            NOTIFY bitfield mask;                         /* Output notification message
            NOCONTROL bitfield mask;                      /* Do not put CR/LF in front of prompt string
        end FLAGS_BITS;
    end FLAGS_OVERLAY;
    FILL_5 byte dimension 3 fill prefix CLIDEF tag $$;  /* Unused
    OUTPID longword unsigned;                             /* PID of subprocess on return
```

```
     LSTSTATUS longword unsigned;                          /* Address to store final subprocess status
     CMDSTR quadword unsigned;                             /* Descriptor of command string
     INPUT quadword unsigned;                              /* Descriptor of input filespec
     OUTPUT quadword unsigned;                             /* Descriptor of output filespec
     PRCNAM quadword unsigned;                             /* Descriptor of name for subprocess
     ASTADR longword unsigned;                             /* Address of termination AST routine
     ASTPRM longword unsigned;                             /* Address of AST routine parameter
     EFN byte unsigned;                                    /* Event flag to set on termination
     VERSION byte unsigned;                                /* Data structure version
     constant SPAWN_VERSION equals 1 prefix CLI$ tag K;    /* Latest version
     constant SPAWN_VERSION equals 1 prefix CLI$ tag C;    /* Latest version
     FILL_6 byte dimension 2 fill prefix CLIDEF tag $$;    /* Unused
     PROMPT quadword unsigned;                             /* Descriptor of prompt string
     CLI quadword unsigned;                                /* Descriptor of cli name
     TABLE quadword unsigned;                              /* Descriptor of cli table name
/*
/* Define the length of the longest "supervisor-mode service" request block,
/* so that programs can allocate a fixed amount of space for the block.
/*

     constant SRVDESC equals . prefix CLI$ tag K;          /* Length of longest "service" callback
     constant SRVDESC equals . prefix CLI$ tag C;          /* Length of longest "service" callback

/*
/* DEFINE THE PARAMETER QUALIFIER DESCRIPTOR
/*


end CLIDEF5;

aggregate CLIDEF6 structure prefix CLI$;
     QDBLKSIZ byte unsigned;                               /* SIZE OF THE FINAL BLOCK
     QDCODE byte unsigned;                                 /* ID CODE FOR THE QUALIFIER
     QDFLGS_OVERLAY union fill;
         QDFLGS byte unsigned;                             /* FLAGS BYTE
         QDFLGS_BITS structure fill;
             ALLOCCUR bitfield mask;                       /* TAKE ACTION ON ALL OCCURANCES
             QDUSRV bitfield mask;                         /* USER CONTEX VALUE IS PRESENT
             QDEXPA bitfield mask;                         /* TAKE ACTION ON EXPLICIT OCCURANCES
         end QDFLGS_BITS;
     end QDFLGS_OVERLAY;
     QDSTAT_OVERLAY union fill;
         QDSTAT byte unsigned;                             /* QUALIFIER STATUS
         QDSTAT_BITS structure fill;
             QUALTRU bitfield mask;                        /* QUALIFIER IS TRUE
             QUALEXP bitfield mask;                        /* QUALIFIER EXPLICITLY STATED
         end QDSTAT_BITS;

     end QDSTAT_OVERLAY;
     QDVALDESC_OVERLAY union fill;
         QDVALDESC quadword unsigned;                      /* QUALIFIER VALUE DESCRIPTOR
         QDVALDESC_FIELDS structure fill;
             QDVALSIZ word unsigned;                       /* SIZE OF VALUE
             FILL_7 word fill prefix CLIDEF tag $$;        /* SPARE WORD
             QDVALADR address;                             /* ADDRESS OF VALUE STRING
```

```
        end QDVALDESC_FIELDS;
    end QDVALDESC_OVERLAY;
    TRUACT address;                                     /* QUALIFIER TRUE ACTION ROUTINE
    FLSACT address;                                     /* QUALIFIER FALSE ACTION ROUTINE
    constant QUALDEF equals . prefix CLI$ tag K;        /* SIZE OF FIXED PART OF STRUCTURE
    constant QUALDEF equals . prefix CLI$ tag C;        /* SIZE OF FIXED PART OF STRUCTURE
    constant QDBITS equals . prefix CLI$ tag K;         /* START OF BIT LISTS(VARIALBLE LENGTH)
    constant QDBITS equals . prefix CLI$ tag C;         /* START OF BIT LISTS(VARIALBLE LENGTH)
    USRVAL longword unsigned;                           /* OPTION USER VALUE IF PRESENT

/*
/* DEFINE SPACE FOR THE RESULT PARSE WORK AREA
/*


end CLIDEF6;

aggregate CLIDEF7 union prefix CLI$;
    WORKAREA longword unsigned dimension 32;            /* ALLOCATE 32 LONG WORDS
    constant WORKAREA equals . prefix CLI$ tag K;       /* SIZE OF HEADER
    constant WORKAREA equals . prefix CLI$ tag C;       /* SIZE OF HEADER


/*
/* DEFINE CLI UTILITY REQUEST CODES
/*
/*          CODES ARE 8 BITS, CONSISTING OF 2 4 BIT FIELDS
/*          THE LEAST 4 BITS ARE SUBFUCTION DEFINITONS
/*          AND THE MOST SIGNIFICANT 4 BITS ARE REQUEST TYPE
/*
                                                        /* DEFINE REQUEST TYPE CODES
    constant(
          UTILOPR                                       /* UTILITY OPERATIONS
        , INPSPEC                                       /* REQUEST FOR AN INPUT SPECIFICATION
        , OUTSPEC                                       /* AN OUTPUT FILE SPECIFICATION
        , PARDONE                                       /* PARAMETER DONE REQUEST
        , VALCONV                                       /* REQUEST A VALUE CONVERSION
        , CLINT                                         /* CLINT OPERATIONS
        ) equals 0 increment 1  prefix CLI tag $K;
/*
/* DEFINE COMPLETE CODES FOR UTILITY OPERATIONS
/*
    constant(
          INITPRS                                       /* REQUEST INITIALIZATION OF PARSE
        , GETCMD                                        /* GET COMMAND BUFFER LIMITS
        , GETQUAL                                       /* OPTAIN STATE OF QUALIFIERS
        , GETOPT                                        /* DECODE COMMAND OPTION
        , GETLINE                                       /* GET COMMAND LINE
/*
/* THE CLISERV REQUEST TYPE APPEARS HERE, INSTEAD OF WITH THE OTHER REQUEST
/* TYPES, BECAUSE IT HAS NO SUBFUNCTIONS ASSOCIATED WITH IT, AND BECAUSE A
/* DAY 1 CODING ERROR CAUSES DCL TO EXPECT THAT THIS REQUEST NUMBER WILL
/* APPEAR IN THE SUBFUNCTION BITS.  THIS MEANS THAT NO UTILITY OPERATION
/* CAN BE DEFINED WITH THE SUBFUNCTION NUMBER 5.
/*
        , CLISERV                                       /* REQUEST A SERVICE FROM THE CLI
        ) equals (CLI$K_UTILOPR@4) increment 1  prefix CLI tag $K;

/*
/* DEFINE COMPLETE CODES FOR INPUT SPECIFICATIONS
/*
    constant(
          INPUT1                                      /* PRIMARY INPUT
        , INPUT2                                      /* SECONDARY INPUT
        , INPUT3                                      /* THIRD,
        , INPUT4                                      /* ETC,ETC,ETC
        ) equals (CLI$K_INPSPEC@4) increment 1  prefix CLI tag $K;
/*
/* DEFINE COMPLETE CODES FOR OUTPUT SPECIFICATIONS
/*
    constant(
          OUTPUT1                                     /* FIRST OUTPUT
        , OUTPUT2                                     /* SECOND OUTPUT
        , OUTPUT3                                     /* THIRD,-
        , OUTPUT4                                     /* ETC,ETC,ETC
        ) equals (CLI$K_OUTSPEC@4) increment 1  prefix CLI tag $K;
/*
/* DEFINE CODES FOR RESULT PARSE PARAMETER COMPLETION
/*
    constant(
          ENDPRM1                                     /* COMPLETED PARAMETER SET 1
        , ENDPRM2                                     /* COMPLETED PARAMETER SET 2
        , ENDPRM3                                     /* COMPLETED PARAMETER SET 3
        , ENDPRM4                                     /* COMPLETED PARAMETER SET 4
        ) equals (CLI$K_PARDONE@4) increment 1  prefix CLI tag $K;
/*
/* DEFINE CODES FOR VALUE CONVERSION REQUESTS
/*
    constant(
          NUMERVAL                                    /* NUMBERIC VALUE
        , ASCIIVAL                                    /* ASCII VALUE
        , KEYWORD                                     /* KEYWORD VALUE
        , KEYVAL                                      /* KEYWORD WITH VALUE
        , FILSPEC                                     /* VALUE IS A FILESPEC
        ) equals (CLI$K_VALCONV@4) increment 1  prefix CLI tag $K;
/*
/* DEFINE COMPLETE CODES FOR UTILITY OPERATIONS
/*
    constant(
          PRESENT                                     /* DETERMINE IF ENTITY IS PRESENT
        , GETVALUE                                    /* GET VALUE OF ENTITY
        , ENDPARSE                                    /* CLEAN UP AFTER PARSING COMMAND
        , DCLPARSE                                    /* PARSE USER COMMAND LINE
        , DISPATCH                                    /* DISPATCH TO ACTION ROUTINE
        , NEXTQUAL                                    /* PROCESS NEXT QUALIFIER
        ) equals (CLI$K_CLINT@4) increment 1  prefix CLI tag $K;
end CLIDEF7;

end_module $CLIDEF;

module $CLISERVDEF;
```

```
/*
/* DEFINE CLI SERVICE REQUEST CODES
/*

                                                        /* DEFINE CLI SERVICE CODES
constant(
      PAUSE                                             /* PAUSE THE IMAGE
    , DEFLOCAL                                          /* DEFINE A SYMBOL IN THE LOCAL TABLE
    , DEFGLOBAL                                         /* DEFINE A SYMBOL IN THE GLOBAL TABLE
    , CHAIN                                             /* PASS AN IMAGE TO RUN AFTER THIS ONE
    , COMMAND                                           /* PASS A COMMAND LINE TO LATER EXECUTE
    , CREALOG                                           /* DEFINE A PROCESS LOGICAL NAME
    , DELELOG                                           /* DELETE A PROCESS LOGICAL NAME
    , DISACTRLY                                         /* DISABLE DCL CONTROL Y PROCESSING
    , ENABCTRLY                                         /* ENABLE DCL CONTROL Y PROCESSING
    , GETSYM                                            /* RETURN VALUE OF A SYMBOL
    , DELELCL                                           /* DELETE A LOCAL SYMBOL
    , DELEGBL                                           /* DELETE A GLOBAL SYMBOL
    , DISAOOB                                           /* DISABLE OUT-OF-BAND CHARACTER(S)
    , ENABOOB                                           /* RE-ENABLE OUT-OF-BAND CHARACTER(S)
    , SPAWN                                             /* SPAWN A SUBPROCESS
    , ATTACH                                            /* ATTACH TO A PROCESS
    ) equals 1 increment 1  prefix CLI tag $K;
/*
/* Define local/global symbol flag returned by GETSYM
/*
constant(
      LOCAL_SYM                                         /* Local symbol
    , GLOBAL_SYM                                        /* Global symbol
    ) equals 1 increment 1  prefix CLI tag $K;

end_module $CLISERVDEF;

module $CLIVERBDEF;
/*
/* DEFINE GENERIC CODES FOR VERBS
/*

                                                        /* DEFINE VERB QENERIC CODES
constant(
      ALLO                                              /* ALLOCATE
    , ANAL                                              /* ANALIZE
    , ASSI                                              /* ASSIGN
    , BASI                                              /* BASIC
    , BLIS                                              /* BLISS
    , COBO                                              /* COBOL
    , CONT                                              /* CONTINUE
    , COPY                                              /* COPY
    , CREA                                              /* CREATE
    , DATA                                              /* DATA
    , DEAL                                              /* DEALLOCATE
    , DEAS                                              /* DEASSIGN
    , DEBU                                              /* DEBUG
    , DEFI                                              /* DEFINE
    , DELE                                              /* DELETE
    , DEPO                                              /* DEPOSIT
    , DIFF                                                /* DIFFERENCE COMMAND
    , DIRE                                                /* DIRECTORY
    , DISM                                                /* DISMOUNT
    , EDIT                                                /* EDIT
    , EOD                                                 /* EOD
    , EXAM                                                /* EXAMINE
    , EXIT                                                /* EXIT
    , FORT                                                /* FORTRAN
    , GOTO                                                /* GOTO
    , HELP                                                /* HELP
    , IF                                                  /* IF
    , INIT                                                /* INITIALIZE
    , INQU                                                /* INQUIRE
    , LINK                                                /* LINK
    , LOGO                                                /* LOGOUT
    , MACR                                                /* MACRO
    , MCR                                                 /* MCR
    , ON                                                  /* ON
    , PRIN                                                /* PRINT
    , RUN                                                 /* RUN
    , SET                                                 /* SET
    , SHOW                                                /* SHOW
    , STAR                                                /* START
    , STOP                                                /* STOP
    , SUBM                                                /* SUBMIT
    , TYPE                                                /* TYPE
    , MOUN                                                /* MOUNT
    , PATC                                                /* PATCH
    , REPL                                                /* REPLAY
    , UNLO                                                /* UNLOCK
    , APPE                                                /* APPEND COMMAND
    , DUMP                                                /* DUMP
    , PURG                                                /* PURGE
    , RENA                                                /* RENAME
    , CANC                                                /* CANCEL
    , LIBR                                                /* LIBRARY
    , SORT                                                /* SORT
    , REQU                                                /* REQUEST
    , SYNC                                                /* SYNCRONIZE
    , CORA                                                /* CORAL
    , PASC                                                /* PASCAL
    , PLI                                                 /* PL/1
    , MESS                                                /* MESSAGE
    ) equals 1 increment 1  prefix CLI$K_VE tag RB;
                                                          /* DEFINE VERB QENERIC CODES
constant(
    FORE                                                  /* FOREIGN COMMAND
    ) equals 255 increment -1  prefix CLI$K_VE tag RB;

end_module $CLIVERBDEF;
```

```
module $CLSDEF;

/*+
/*
/* Security classification mask block. Contains security and integrity
/* level and categories for non-discretionary access controls.
/*
/*-

aggregate CLSDEF structure prefix CLS$;
    SECUR_LEV byte unsigned;              /* Security level
    INTEG_LEV byte unsigned;              /* Integrity level
    FILL_T word fill;                     /* Reserved
    SECUR_CAT quadword;                   /* Security category mask
    INTEG_CAT quadword;                   /* Integrity category mask
end CLSDEF;

end_module $CLSDEF;
```

```
module $CRDEF;
/*+
/* CARD READER STATUS BITS
/*-


aggregate CRDEF  union prefix CR$;
    CRDEF_BITS structure fill;
        TMODE bitfield mask length 4;                   /* TRANSLATION MODE
    end CRDEF_BITS;

/*
/* TRANSLATION MODE DEFINITIONS
/*

                                                        /*
    constant T026       equals 0  prefix CR tag $K;     /*026 PUNCH CODE TRANSLATION
    constant T029       equals 1  prefix CR tag $K;     /*029 PUNCH CODE TRANSLATION

end CRDEF;

end_module $CRDEF;
```

```
module $DCDEF;
/*
/* DEVICE ADAPTER, CLASS, AND TYPE DEFINITIONS
/*

/*
/* DEFINE ADAPTER TYPES
/*
                                                          /*DEFINE ADAPTER TYPES
constant MBA    equals 0  prefix AT tag $;                /* MASSBUS ADAPTER
constant UBA    equals 1  prefix AT tag $;                /* UNIBUS ADAPTER
constant DR     equals 2  prefix AT tag $;                /* DR32 ADAPTER
constant MPM    equals 3  prefix AT tag $;                /* MULTI-PORT MEMORY
constant CI     equals 4  prefix AT tag $;                /* CI BUS
constant NULL   equals 5  prefix AT tag $;                /* NULL (SOFTWARE) ADAPTER
/*
/* DEFINE DEVICE CLASSES
/*
                                                          /*DEFINE DEVICE CLASSES
constant DISK   equals 1  prefix DC tag $;                /* DISK
constant TAPE   equals 2  prefix DC tag $;                /* TAPES
constant SCOM   equals 32  prefix DC tag $;               /* SYNCHRONOUS COMMUNICATIONS DEVICES
constant CARD   equals 65  prefix DC tag $;               /* CARD READER
constant TERM   equals 66  prefix DC tag $;               /* TERMINAL
constant LP     equals 67  prefix DC tag $;               /* LINE PRINTER
constant WORKSTATION equals 70 prefix DC tag $;           /* WORKSTATIONS
constant REALTIME         equals 96  prefix DC tag $;     /* REAL-TIME
constant BUS    equals 128  prefix DC tag $;              /* BUSES, E.G, CI
constant MAILBOX          equals 160  prefix DC tag $;    /* MAILBOX
constant JOURNAL          equals 161  prefix DC tag $;    /* JOURNAL
constant MISC   equals 200  prefix DC tag $;              /* MISCELLANEOUS DEVICES

/*
/* DEFINE DEVICE TYPES
/*

                                                          /*

/*
/* DISK DEVICES
/*

constant RK06   equals 1  prefix DT tag $;                /*RK06 DISK
constant RK07   equals 2  prefix DT tag $;                /*RK07 DISK
constant RP04   equals 3  prefix DT tag $;                /*RP04 DISK
constant RP05   equals 4  prefix DT tag $;                /*RP05 DISK
constant RP06   equals 5  prefix DT tag $;                /*RP06 DISK
constant RM03   equals 6  prefix DT tag $;                /*RM03 DISK
constant RP07   equals 7  prefix DT tag $;                /*RP07 DISK
constant RP07HT equals 8  prefix DT tag $;                /*RP07 DISK WITH HEAD/TRACK
constant RL01   equals 9  prefix DT tag $;                /*RL01 DISK
constant RL02   equals 10  prefix DT tag $;               /*RL02 DISK
constant RX02   equals 11  prefix DT tag $;               /*RX02 DISK
constant RX04   equals 12  prefix DT tag $;               /*RX04 DISK
constant RM80   equals 13  prefix DT tag $;               /*RM80 DISK
constant TU58    equals 14  prefix DT tag $;              /*TU58
constant RM05    equals 15  prefix DT tag $;              /*RM05 DISK
constant RX01    equals 16  prefix DT tag $;              /*RX01 DISK
constant ML11    equals 17  prefix DT tag $;              /*ML11 disk
constant RB02    equals 18  prefix DT tag $;              /*R02 ON RB730
constant RB80    equals 19  prefix DT tag $;              /*R80 ON RB730
constant RA80    equals 20  prefix DT tag $;              /*R80 ON INTELLIGENT CONTROLLER
constant RA81    equals 21  prefix DT tag $;              /*R81 ON INTELLIGENT CONTROLLER
constant RA60    equals 22  prefix DT tag $;              /*PINON ON INTELLIGENT CONTROLLER
constant RZ01    equals 23  prefix DT tag $;              /*AZTEC REMOVABLE (Old name)
constant RC25    equals 23  prefix DT tag $;              /*AZTEC REMOVABLE (New name)
constant RZF01   equals 24  prefix DT tag $;              /*AZTEC FIXED (Old name)
constant RCF25   equals 24  prefix DT tag $;              /*AZTEC FIXED (New name)
constant RD51    equals 25  prefix DT tag $;              /*RD51 FIXED DISK DRIVE
constant RX50    equals 26  prefix DT tag $;              /*RX50 FLOPPY DISK DRIVE
constant RD52    equals 27  prefix DT tag $;              /*RD52 FIXED DISK DRIVE
constant RD53    equals 28  prefix DT tag $;              /*RD53 FIXED DISK DRIVE
constant RD26    equals 29  prefix DT tag $;              /*RD26 FIXED DISK DRIVE
constant RA82    equals 30  prefix DT tag $;              /*RA82 FIXED DISK DRIVE
constant RC26    equals 31  prefix DT tag $;              /*AZTEC II REMOVABLE
constant RCF26   equals 32  prefix DT tag $;              /*AZTEC II FIXED
constant CRX50   equals 33  prefix DT tag $;              /*Console RX50
constant CDR50   equals 34  prefix DT tag $;              /*CDR50
constant RX31    equals 35  prefix DT tag $;              /*RX31
constant RX32    equals 36  prefix DT tag $;              /*RX32
constant RX18    equals 37  prefix DT tag $;              /*RX18
{ Add device type definitions for DIGITAL manufactured disks above this line. }

constant FD1     equals 129 prefix DT tag $;              /*FOREIGN DISK TYPE 1
constant FD2     equals 130 prefix DT tag $;              /*FOREIGN DISK TYPE 2
constant FD3     equals 131 prefix DT tag $;              /*FOREIGN DISK TYPE 3
constant FD4     equals 132 prefix DT tag $;              /*FOREIGN DISK TYPE 4
constant FD5     equals 133 prefix DT tag $;              /*FOREIGN DISK TYPE 5
constant FD6     equals 134 prefix DT tag $;              /*FOREIGN DISK TYPE 6
constant FD7     equals 135 prefix DT tag $;              /*FOREIGN DISK TYPE 7
constant FD8     equals 136 prefix DT tag $;              /*FOREIGN DISK TYPE 8

/*
/* TAPE DEVICES
/*

constant TE16    equals 1   prefix DT tag $;              /*TE16 MAGTAPE
constant TU45    equals 2   prefix DT tag $;              /*TU45 MAGTAPE
constant TU77    equals 3   prefix DT tag $;              /*TU77 MAGTAPE
constant TS11    equals 4   prefix DT tag $;              /*TS11 MAGTAPE
constant TU78    equals 5   prefix DT tag $;              /*TU78 MAGTAPE
constant TA78    equals 6   prefix DT tag $;              /*TA78 MAGTAPE
constant TU80    equals 7   prefix DT tag $;              /*TU80 MAGTAPE
constant TU81    equals 8   prefix DT tag $;              /*TU81 MAGTAPE
constant TA81    equals 9   prefix DT tag $;              /*TA81 MAGTAPE
constant TK50    equals 10  prefix DT tag $;              /*TK50 CARTRIDGE TAPE
/*
/* TERMINAL DEVICE TYPES
/*
/* new definitions for terminal types should be placed in $ttdef only
/* this table remains around for compatibility only
/*
        /* **** MATCHES $TTDEF *****
constant TTYUNKN        equals 0  prefix DT tag $;          /* UNKNOWN TERMINAL
constant VT05   equals 1  prefix DT tag $;                  /* VT05
constant FT1    equals 16  prefix DT tag $;                 /* FOREIGN TERMINAL TYPES
constant FT2    equals 17  prefix DT tag $;
constant FT3    equals 18  prefix DT tag $;
constant FT4    equals 19  prefix DT tag $;
constant FT5    equals 20  prefix DT tag $;
constant FT6    equals 21  prefix DT tag $;
constant FT7    equals 22  prefix DT tag $;
constant FT8    equals 23  prefix DT tag $;                 /* END OF FOREIGN TYPES
                                                            /* RESERVE REST UP TO 32 FOR EXTENSIONS
constant LAX    equals 32  prefix DT tag $;                 /* LA TYPE TERMINAL
constant LA36   equals 32  prefix DT tag $;                 /* LA36
constant LA120  equals 33  prefix DT tag $;
constant VT5X   equals 64  prefix DT tag $;                 /* VT5X TYPE
constant VT52   equals 64  prefix DT tag $;                 /* VT52
constant VT55   equals 65  prefix DT tag $;                 /* VT55
constant TQ_BTS equals 4   prefix DT tag $;                 /* TQ_BTS
constant TER401X equals 10 prefix DT tag $;                 /* TER401X series
constant VT100  equals 96  prefix DT tag $;                 /* VT100
constant VK100  equals 2  prefix DT tag $;
constant VT173  equals 3  prefix DT tag $;
constant LA34   equals 34  prefix DT tag $;
constant LA38   equals 35  prefix DT tag $;
constant LA12   equals 36  prefix DT tag $;
constant LA24   equals 37  prefix DT tag $;
constant LA100  equals 37  prefix DT tag $;
constant LQP02  equals 38  prefix DT tag $;
constant VT101  equals 97  prefix DT tag $;
constant VT102  equals 98  prefix DT tag $;
constant VT105  equals 99  prefix DT tag $;
constant VT125  equals 100  prefix DT tag $;
constant VT131  equals 101  prefix DT tag $;
constant VT132  equals 102  prefix DT tag $;
constant DZ11   equals 66  prefix DT tag $;                 /* DZ11 CONTROLLER
constant DZ32   equals 67  prefix DT tag $;                 /* DZ32 CONTROLLER
constant DZ730  equals 68  prefix DT tag $;                 /* DZ730 (COMBO) CONTROLLER
constant DMZ32  equals 69  prefix DT tag $;                 /* DMZ32 CONTROLLER
constant DHV    equals 70  prefix DT tag $;                 /* DHV CONTROLLER
constant DHU    equals 71  prefix DT tag $;                 /* DHU CONTROLLER
/*
/* Terminal WORKSTATIONS
/*
constant VS100 equals 1 prefix DT tag $;                    /* VAXstation 100
constant VS125 equals 2 prefix DT tag $;                    /* VAXstation 125
constant VS300 equals 3 prefix DT tag $;                    /* VAXstation 300
constant VD equals 4 prefix DT tag $;                       /* VAXstation Vir. Device
/*
/* SYNCHRONOUS COMMUNICATIONS DEVICE TYPES
/*
constant DMC11  equals 1  prefix DT tag $;                  /* DMC11
constant DMR11  equals 2  prefix DT tag $;                  /* DMR11
constant XK_3271        equals 3  prefix DT tag $;          /* DUP-11 FOR 3271 PROTOCOL EMULATOR
constant XJ_2780               equals 4  prefix DT tag $;          /* DUP-11 FOR 2780 " "
constant NW_X25 equals 5  prefix DT tag $;                         /* X25 PROTOCOL EMULATOR
constant NV_X29 equals 6  prefix DT tag $;                         /* X29 " "
constant SB_ISB11              equals 7  prefix DT tag $;          /* ISB-11 DEC dataway
constant MX_MUX200             equals 8  prefix DT tag $;          /* MUX-200 PROTOCOL EMULATOR
constant DMP11  equals 9  prefix DT tag $;                         /* DMP11
constant DMF32  equals 10  prefix DT tag $;                        /* DMF32
constant XV_3271               equals 11  prefix DT tag $;         /* DV-11 3271 PROTOCOL EMULATOR
constant CI     equals 12  prefix DT tag $;                        /* CI - Computer Interconnect
constant NI     equals 13  prefix DT tag $;                        /* NI - Network Interconnect
constant UNA11  equals 14  prefix DT tag $;                        /* UNIBUS to NI adapter
constant DEUNA  equals 14  prefix DT tag $;                        /* UNIBUS to NI adapter
constant YN_X25 equals 15  prefix DT tag $;                        /* KMS11 X.25 P. E.
constant YO_X25 equals 16  prefix DT tag $;                        /* " " "
constant YP_ADCCP              equals 17  prefix DT tag $;         /* " ADCCP P.E.
constant YQ_3271               equals 18  prefix DT tag $;         /* " " "
constant YR_DDCMP              equals 19  prefix DT tag $;         /* " DDCMP
constant YS_SDLC               equals 20  prefix DT tag $;         /* " SDLC
constant UK_KTC32              equals 21  prefix DT tag $;         /* " KTC32
constant DEQNA  equals 22  prefix DT tag $;                        /* Q-BUS to NI adapter
constant DMV11  equals 23  prefix DT tag $;                        /* DMV11
constant LANCE  equals 24  prefix DT tag $;                        /* SCORPIO to NI adapter
constant DELUA  equals 25  prefix DT tag $;                        /* LSI version of DEUNA
constant NQ_3271 equals 26  prefix DT tag $;                       /* DHCF

/*
/* LINE PRINTER AND CARD READER DEVICE TYPES
/*
constant LP11    equals 1  prefix DT tag $;                        /* LP11
constant LA11    equals 2  prefix DT tag $;                        /* LA11
constant LA180   equals 3  prefix DT tag $;                        /* LA180

constant CR11    equals 1  prefix DT tag $;                        /* CR11 CARD READER

/*
/* MAILBOX DEVICE TYPES
/*
constant MBX     equals 1  prefix DT tag $;                        /* LOCAL MEMORY MAILBOX
constant SHRMBX  equals 2  prefix DT tag $;                        /* SHARED MEMORY MAILBOX
constant NULL    equals 3  prefix DT tag $;                        /* The NULL DEVICE

/*
/* REALTIME DEVICE TYPES
/*
constant LPA11   equals 1  prefix DT tag $;                        /* LPA-11
constant DR780   equals 2  prefix DT tag $;                        /* DR780
constant DR750   equals 3  prefix DT tag $;                        /* DR750
constant DR11W   equals 4  prefix DT tag $;                        /* DR11W
constant PCL11R  equals 5  prefix DT tag $;                        /* PCL11 RECEIVER (CSS)
constant PCL11T  equals 6  prefix DT tag $;                        /* PCL11 TRANSMTTER (CSS)
constant DR11C   equals 7  prefix DT tag $;                        /* DR11C PARALLEL INTERFACE
constant BS_DT07               equals 8  prefix DT tag $;          /* UNIBUS SWITCH
constant XP_PCL11B             equals 9  prefix DT tag $;          /* PCL-11B (DECNET and NONDECNET mode CSS)
constant IX_IEX11              equals 10  prefix DT tag $;         /* IEEE-488 to UNIBUS INTERFACE
constant FP_FEPCM              equals 11  prefix DT tag $;         /* FEPCM CSS front processor
constant TK_FCM                equals 12  prefix DT tag $;         /* FEPCM CSS front processor
constant XI_DR11C         equals 13  prefix DT tag $;     /* PARALLEL INTERFACE ON DMF-32

/*
/* BUS CLASS DEVICES
/*

constant CI780  equals 1  prefix DT tag $;                /* CI780
constant CI750  equals 2  prefix DT tag $;                /* CI750
constant UQPORT equals 3  prefix DT tag $;                /* UQPORT is generic UDA
constant UDA50  equals 3  prefix DT tag $;                /* UDA50
constant UDA50A equals 4  prefix DT tag $;                /* UDA50A
constant LESI   equals 5  prefix DT tag $;                /* Low end storage
constant TU81P  equals 6  prefix DT tag $;                /* TU81 port
constant RDRX   equals 7  prefix DT tag $;                /* RDRX port
constant TK50P  equals 8  prefix DT tag $;                /* TK50 port
constant RUX50P equals 9  prefix DT tag $;                /* RUX50 port
constant RC26P  equals 10 prefix DT tag $;                /* RC26P port
constant QDA50  equals 11 prefix DT tag $;                /* QDA50 port
constant BDA50  equals 12 prefix DT tag $;                /* BDA50 port
constant CDR50P equals 13 prefix DT tag $;                /* CDR50 port
constant QDA25  equals 14 prefix DT tag $;                /* QDA25 port

/*
/* JOURNAL DEVICES
/*
constant UNKNJNL          equals 0  prefix DT tag $;      /* UNKNOWN JOURNAL TYPE (ONLY IN TEMPLATE)
constant RUJNL  equals 1  prefix DT tag $;                /* RECOVERY UNIT JOURNAL
constant BIJNL  equals 2  prefix DT tag $;                /* BEFORE IMAGE JOURNAL
constant AIJNL  equals 3  prefix DT tag $;                /* AFTER IMAGE JOURNAL
constant ATJNL  equals 4  prefix DT tag $;                /* AUDIT TRAIL JOURNAL
constant CLJNL  equals 5  prefix DT tag $;                /* CONTROL JOURNAL

/*
/* MISCELLANEOUS DEVICES
/*
constant DN11   equals 1  prefix DT tag $;                /* AUTODIALER

end_module $DCDEF;
```

```
module $DEVDEF;
/*
/*  THE FOLLOWING BITS DEFINE THE DEVICE CHARACTERISTICS FOR
/*  BOTH THE UCBS AND RMS.
/*

aggregate DEVDEF  union prefix DEV$;
    DEVDEF_BITS0 structure fill;
        REC bitfield mask;                              /* DEVICE RECORD ORIENTED
        CCL bitfield mask;                              /* CARRIAGE CONTROL DEVICE
        TRM bitfield mask;                              /* DEVICE IS A TERMINAL
        DIR bitfield mask;                              /* DEVICE IS DIRECTORY STRUCTURED
        SDI bitfield mask;                              /* DEVICE IS SINGLE DIRECTORY STRUCTURED
        SQD bitfield mask;                              /* SEQUENTIAL BLOCK-ORIENTED DEVICE (I.E., MAGTAPE)
        SPL bitfield mask;                              /* DEVICE BEING SPOOLED
        OPR bitfield mask;                              /* DEVICE IS AN OPERATOR
        RCT bitfield mask;                              /* DISK CONTAINS RCT (DEC STANDARD 166 DISK)
        FILL_1 bitfield length 4 fill prefix DEVDEF tag $$;/* SPARES TO CORRESPOND WITH RSX11M
        NET bitfield mask;                              /* NETWORK DEVICE
        FOD bitfield mask;                              /* FILES-ORIENTED DEVICE (I.E., DISK AND MT)
        DUA bitfield mask;                              /* DEVICE IS DUAL PORTED
        SHR bitfield mask;                              /* DEVICE SHAREABLE
        GEN bitfield mask;                              /* DEVICE IS A GENERIC DEVICE
        AVL bitfield mask;                              /* DEVICE AVAILABLE FOR USE
        MNT bitfield mask;                              /* DEVICE IS MOUNTED
        MBX bitfield mask;                              /* DEVICE IS A MAILBOX
        DMT bitfield mask;                              /* DEVICE MARKED FOR DISMOUNT
        ELG bitfield mask;                              /* DEVICE HAS ERROR LOGGING ENABLED
        ALL bitfield mask;                              /* DEVICE IS ALLOCATED
        FOR bitfield mask;                              /* DEVICE IS MOUNTED FOREIGN (I.E., NON-FILE STRUCTURED)
        SWL bitfield mask;                              /* DEVICE IS SOFTWARE WRITE LOCKED
        IDV bitfield mask;                              /* DEVICE CAPABLE OF PROVIDING INPUT
        ODV bitfield mask;                              /* DEVICE CAPABLE OF PROVIDING OUTPUT
        RND bitfield mask;                              /* DEVICE ALLOWS RANDOM ACCESS
        RTM bitfield mask;                              /* DEVICE IS REALTIME IN NATURE
        RCK bitfield mask;                              /* DEVICE HAS READ CHECKING ENABLED
        WCK bitfield mask;                              /* DEVICE HAS WRITE CHECKING ENABLED
    end DEVDEF_BITS0;
    DEVDEF_BITS1 structure fill;
        CLU bitfield mask;                              /* DEVICE IS AVAILABLE CLUSTER-WIDE
        DET bitfield mask;                              /* DEVICE IS DETACHED TERMINAL
        RTT bitfield mask;                              /* DEVICE HAS REMOTE TERMINAL UCB EXTENSION
        CDP bitfield mask;                              /* DUAL PATH DEVICE WITH 2 UCBs
        "2P" bitfield mask;                             /* TWO PATHS ARE KNOWN TO THIS DEVICE
        MSCP bitfield mask;                             /* DEVICE ACCESSED USING MSCP (disk or tape)
                                                        {  Before using this bit to diferentiate
                                                        {  between types of disk and tape devices,
                                                        {  be sure that no other, more appropriate,
                                                        {  differentiation mechanism exists.
        SSM bitfield mask;                              /* DEVICE IS A SHADOW SET MEMBER
        SRV bitfield mask;                              /* DEVICE IS SERVED VIA THE MSCP SERVER
        RED bitfield mask;                              /* DEVICE IS redirected terminal
        NNM bitfield mask;                              /* DEVICE HAS "node$" PREFIX
    end DEVDEF_BITS1;
end DEVDEF;
```

```
end_module $DEVDEF;
```

```
module $DIBDEF;
/*+
/* DEVICE INFORMATION BLOCK DEFINITIONS
/*-


aggregate DIBDEF structure prefix DIB$;
    DEVCHAR longword unsigned;                          /*DEVICE CHARACTERISTICS
    DEVCLASS byte unsigned;                             /*DEVICE CLASS
    DEVTYPE byte unsigned;                              /*DEVICE TYPE
    DEVBUFSIZ word unsigned;                            /*DEVICE BUFFER SIZE
    DEVDEPEND_OVERLAY union fill;
        DEVDEPEND longword unsigned;                    /*DEVICE DEPENDENT INFORMATION
        DEVDEPEND_FIELDS structure fill;
            SECTORS byte unsigned;                      /*(DISK ONLY) SECTORS PER TRACK
            TRACKS byte unsigned;                       /* '' TRACKS PER CYLINDER
            CYLINDERS word unsigned;                    /* '' NUMBER OF CYLINDERS
        end DEVDEPEND_FIELDS;
    end DEVDEPEND_OVERLAY;
    UNIT word unsigned;                                 /*DEVICE UNIT NUMBER
    DEVNAMOFF word unsigned;                            /*OFFSET TO DEVICE NAME COUNTED STRING
    PID longword unsigned;                              /*DEVICE OWNER PROCESS IDENTIFICATION
    OWNUIC longword unsigned;                           /*DEVICE OWNER USER IDENTIFICATION CODE
    VPROT word unsigned;                                /*DEVICE PROTECTION MASK
    ERRCNT word unsigned;                               /*DEVICE ERROR COUNT
    OPCNT longword unsigned;                            /*DEVICE OPERATIONS COUNT
    VOLNAMOFF word unsigned;                            /*OFFSET TO VOLUME LABEL COUNTED STRING
    RECSIZ word unsigned;                               /*BLOCKED RECORD SIZE
    DEVNAME character length 76;                        /*SPACE FOR DEVNAME AND LABEL (64+12)
    MAXBLOCK longword unsigned;                         /*DISK VOLUME SIZE IN BLOCKS
    constant "LENGTH" equals . prefix DIB$ tag K;       /*LENGTH OF TOTAL BUFFER
    constant "LENGTH" equals . prefix DIB$ tag C;       /*LENGTH OF TOTAL BUFFER

end DIBDEF;

end_module $DIBDEF;
```

```
module $DMPDEF;

/*
/* LAYOUT OF THE HEADER BLOCK OF THE SYSTEM DUMP FILE
/* (WHICH IS THE FIRST DISK BLOCK OF SYS$SYSTEM:SYSDUMP.DMP)
/*


aggregate DMPDEF structure prefix DMP$;
    ERRSEQ longword unsigned;                           /* LAST ERROR LOG SEQ. NUMBER
    FLAGS_OVERLAY union fill;
        FLAGS longword unsigned;                        /* DUMP FILE FLAGS
        FLAGS_FIELDS structure fill;
            FLAGS_OVERLAY1 union;
                FLAGS word unsigned;                    /* NEW FORMAT DUMP: DUMP FILE FLAGS
                FLAGS_BITS structure fill;
                    OLDDUMP bitfield;                   /* SET IF DUMP ALREADY ANALYZED
                    EMPTY bitfield;                     /* SET IF DUMP HAS NO DATA BLOCKS
                end FLAGS_BITS;
            end FLAGS_OVERLAY1;
            DUMPVER word unsigned;                      /* NEW FORMAT DUMP: DUMP FILE VERSION NUMBER
                                                        /* NEW FORMAT DUMP:     0 = PRE-RELEASE 2 FORMAT
                                                        /* NEW FORMAT DUMP:     1 = RELEASE 2.0 FORMAT

        end FLAGS_FIELDS;
    end FLAGS_OVERLAY;
    SBR longword unsigned;                              /* SYSTEM BASE REGISTER
    SLR longword unsigned;                              /* SYSTEM LENGTH REGISTER
    KSP longword unsigned;                              /* KERNEL STACK POINTER
    ESP longword unsigned;                              /* EXECUTIVE STACK POINTER
    SSP longword unsigned;                              /* SUPERVISOR STACK POINTER
    USP longword unsigned;                              /* USER STACK POINTER
    ISP longword unsigned;                              /* INTERRUPT STACK POINTER
    REGS_OVERLAY union fill;
        REGS longword unsigned dimension 14;            /* OLD FORMAT DUMP: R0 - R13
        MEMDSC_OVERLAY union fill;
            MEMDSC longword unsigned;                   /* NEW FORMAT DUMP: 8 MEMORY DESCRIPTORS
            MEMDSC_BITS structure fill;
                PAGCNT bitfield length 24;              /* NEW FORMAT DUMP: ! OF PAGES IN MEMORY
                TR bitfield length 8;                   /* NEW FORMAT DUMP: TR ! FOR MEMORY
                BASEPFN bitfield length 32;             /* NEW FORMAT DUMP: BASE PFN FOR MEMORY
            end MEMDSC_BITS;
            constant MEMDSCSIZ  equals 8  prefix DMP tag $C;/* NEW FORMAT DUMP: SIZE OF ONE MEMORY DESCRPTR
            constant NMEMDSC    equals 8  prefix DMP tag $C;/* NEW FORMAT DUMP: NUMBER OF MEMORY DESCRIPTORS
        end MEMDSC_OVERLAY;
    end REGS_OVERLAY;
end DMPDEF;

aggregate DMPDEF1 structure prefix DMP$;
    FILL_1 byte dimension 92 fill prefix DMPDEF tag $$;
    SP longword unsigned;                               /* OLD FORMAT DUMP: STACK POINTER
    PC longword unsigned;                               /* OLD FORMAT DUMP: PROGRAM COUNTER
    SYSVER_OVERLAY union fill;
        SYSVER longword unsigned;                       /* NEW FORMAT DUMP: SYSTEM VERSION NUMBER
        PSL longword unsigned;                          /* OLD FORMAT DUMP: PROGRAM STATUS LONGWORD
    end SYSVER_OVERLAY;
    CHECK longword unsigned;                            /* NEW FORMAT DUMP: ONES COMPLEMENT OF SYSVER
    constant "LENGTH" equals . prefix DMP$ tag K;          /* LENGTH OF FILE HEADER
    constant "LENGTH" equals . prefix DMP$ tag C;          /* LENGTH OF FILE HEADER
    CRASHERL longword unsigned;                            /* NEW FORMAT DUMP: SYSTEM CRASH ERR LOG ENTRY

end DMPDEF1;

end_module $DMPDEF;
```

```
module $DMTDEF;
/*
/* FLAG BITS FOR THE $DISMOU (DISMOUNT) SYSTEM SERVICE.
/*

aggregate DMTDEF  union prefix DMT$;
    DMTDEF_BITS structure fill;
        NOUNLOAD bitfield mask;                         /* DO NOT UNLOAD (SPIN DOWN) THE VOLUME
        UNIT bitfield mask;                             /* DISMOUNT ONLY THE SPECIFIED VOLUME
        ABORT bitfield mask;                            /* FORCED DISMOUNT
        CLUSTER bitfield mask;                          /* CLUSTER-WIDE DISMOUNT
    end DMTDEF_BITS;

end DMTDEF;

end_module $DMTDEF;
```

```
module $DVIDEF;

/*+
/*
/* Get Device and Volume Information Data Identifier Definitions
/*
/* **** NOTE ****
/*
/*      New items must always be added at the END of the list so that
/*      users will not have to relink.
/*
/*-


constant(
        DEVCHAR             /* Device characteristics - VALUE - 4 bytes
      , DEVCLASS            /* Device class - VALUE - 1 byte
      , DEVTYPE             /* Device type - VALUE - 1 byte
      , DEVBUFSIZ           /* Device buffer size - VALUE - 2 bytes
      , DEVDEPEND           /* Device dependent information - VALUE - 4 bytes
      , UNIT                /* Unit number - VALUE - 2 bytes
      , PID                 /* Process identification of device owner - VALUE - 4 bytes
      , OWNUIC              /* UIC of device owner - VALUE - 4 bytes
      , VPROT               /* Volume protection mask - VALUE - 2 bytes
      , ERRCNT              /* Error count - VALUE - 2 bytes
      , OPCNT               /* Operation count - VALUE - 4 bytes
      , RECSIZ              /* Blocked record size - VALUE - 2 bytes
      , MAXBLOCK            /* Number of logical blocks on the volume (disk) - VALUE - 4 bytes
      , DEVDEPEND2          /* Additional device dependent data - VALUE - 4 bytes
      , REFCNT              /* Reference count of processes - VALUE - 2 bytes
      , DEVNAM              /* Device name - STRING - 64 bytes
      , VOLNAM              /* Volume name - STRING - 12 bytes
      , SECTORS             /* Number of sectors per track (disk) - VALUE - 1 byte
      , TRACKS              /* Number of tracks per cylinder (disk) - VALUE - 1 byte
      , CYLINDERS           /* Number of cylinders on the volume (disk) - VALUE - 2 bytes
      , FREEBLOCKS          /* Number of free blocks on the volume (disk) - VALUE - 4 bytes
      , LOGVOLNAM           /* Logical volume name - STRING - 64 bytes
      , VOLNUMBER           /* Number of this volume in volume set (disk) - VALUE - 4 byte
      , VOLCOUNT            /* Count of volumes in volume set (disk) - VALUE - 4 byte
      , ROOTDEVNAM          /* Device name of root volume in volume set (disk) - STRING - 64 bytes
      , NEXTDEVNAM          /* Device name of next volume in volume set (disk) - STRING - 64 bytes
      , TRANSCNT            /* Volume Transaction Count - VALUE - 2 bytes
      , MOUNTCNT            /* Mount count - VALUE - 2 bytes
      , CLUSTER             /* Volume Cluster Size (disk) - VALUE - 2 bytes
      , MAXFILES            /* Maximum Files on Volume (disk) - VALUE - 4 bytes
      , SERIALNUM           /* Volume Serial Number (disk) - VALUE - 4 bytes
      , ACPPID              /* ACP Process ID - VALUE - 4 bytes
      , ACPTYPE             /* ACP type code - VALUE - 1 byte
      , CONCEALED           /* Device is a concealed device - BOOLEAN - 1 byte
/*
/** THE FOLLOWING CODES ARE THE INDIVIDUAL BITS OF THE DEVCHAR LONGWORD
/*
      , REC                 /* DEVICE RECORD ORIENTED
      , CCL                 /* CARRIAGE CONTROL DEVICE
      , TRM                 /* DEVICE IS A TERMINAL
```

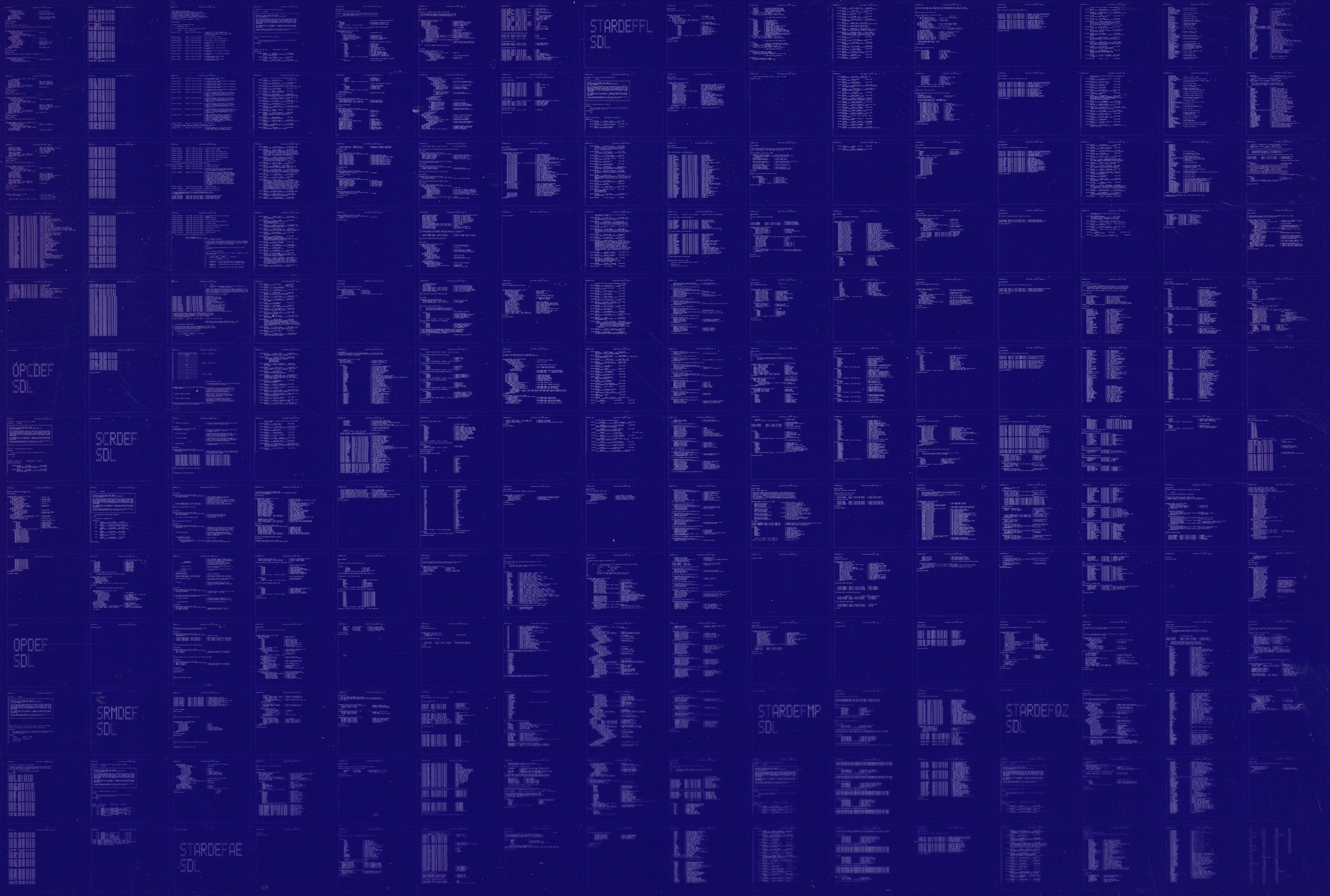
```
    , DIR                /* DEVICE IS DIRECTORY STRUCTURED
    , SDI                /* DEVICE IS SINGLE DIRECTORY STRUCTURED
    , SQD                /* SEQUENTIAL BLOCK-ORIENTED DEVICE (I.E., MAGTAPE)
    , SPL                /* DEVICE BEING SPOOLED
    , OPR                /* DEVICE IS AN OPERATOR
    , RCT                /* DISK CONTAINS RCT (DEC STANDARD 166 DISK)
    , NET                /* NETWORK DEVICE
    , FOD                /* FILES-ORIENTED DEVICE (I.E., DISK AND MT)
    , DUA                /* DEVICE IS DUAL PORTED
    , SHR                /* DEVICE SHAREABLE
    , GEN                /* DEVICE IS A GENERIC DEVICE
    , AVL                /* DEVICE AVAILABLE FOR USE
    , MNT                /* DEVICE IS MOUNTED
    , MBX                /* DEVICE IS A MAILBOX
    , DMT                /* DEVICE MARKED FOR DISMOUNT
    , ELG                /* DEVICE HAS ERROR LOGGING ENABLED
    , ALL                /* DEVICE IS ALLOCATED
    , FOR                /* DEVICE IS MOUNTED FOREIGN (I.E., NON-FILE STRUCTURED)
    , SWL                /* DEVICE IS SOFTWARE WRITE LOCKED
    , IDV                /* DEVICE CAPABLE OF PROVIDING INPUT
    , ODV                /* DEVICE CAPABLE OF PROVIDING OUTPUT
    , RND                /* DEVICE ALLOWS RANDOM ACCESS
    , RTM                /* DEVICE IS REALTIME IN NATURE
    , RCK                /* DEVICE HAS READ CHECKING ENABLED
    , WCK                /* DEVICE HAS WRITE CHECKING ENABLED
/*
/**  THE FOLLOWING CODES ARE THE INDIVIDUAL BITS OF THE DEVDEPEND LONGWORD
/**  (AS DEFINED FOR TERMINALS: TTDEF IN STARDEFQZ.SDL)
/*
    , TT_PASSALL
    , TT_NOECHO
    , TT_NOTYPEAHD
    , TT_ESCAPE
    , TT_HOSTSYNC
    , TT_TTSYNC
    , TT_SCRIPT
    , TT_LOWER
    , TT_MECHTAB
    , TT_WRAP
    , TT_CRFILL
    , TT_LFFILL
    , TT_SCOPE
    , TT_REMOTE
    , TT_EIGHTBIT
    , TT_MBXDSABL
    , TT_NOBRDCST
    , TT_READSYNC
    , TT_MECHFORM
    , TT_HALFDUP
    , TT_MODEM
    , TT_OPER
    , TT_PAGE
/*
/**  THE FOLLOWING CODES ARE THE INDIVIDUAL BITS OF THE DEVDEPEND2 LONGWORD
/**  (AS DEFINED FOR TERMINALS: TT2DEF IN STARDEFQZ.SDL)
/*
```

```
    , TT_LOCALECHO
    , TT_AUTOBAUD
    , TT_HANGUP
    , TT_MODHANGUP
    , TT_BRDCSTMBX
    , TT_DMA
    , TT_ALTYPEAHD
    , TT_SETSPEED
    , TT_DCL_MAILBX                             /* TEMP DEFINITIONS FOR DCL SPAWN
    , TT_EDITING
    , TT_INSERT
    , TT_FALLBACK
    , TT_DIALUP
    , TT_SECURE
    , TT_DISCONNECT
    , TT_PASTHRU
    , TT_SIXEL
    , TT_DRCS
    , TT_PRINTER
    , TT_APP_KEYPAD
    , TT_SYSPWD
    , TT_ANSICRT
    , TT_REGIS
    , TT_BLOCK
    , TT_AVO
    , TT_EDIT
    , TT_DECCRT
/*
/**  THE FOLLOWING CODES ARE REGULAR ITEMS
/*
    , STS               /* STATUS LONGWORD
    , DEVSTS            /* DEVICE STATUS WORD
    , DEVCHAR2          /* Second device characteristics longword - VALUE - 4 bytes
    , FULLDEVNAM        /* Fully qualified device name
    , LOCKID            /* Device lock id - VALUE - 4 bytes
    , ALLDEVNAM         /* Allocation class + device name
    , VOLSETMEM         /* Volume set member
    , DEVLOCKNAM        /* Device lock name
/*
/** THE FOLLOWING CODES SUPPORT FEATURES OF DUAL-PATH AND SHADOW-SET DEVICES
/*
    , ALLOCLASS         /* Allocation class of host(s)
    , ALT_HOST_AVAIL    /* Alternate host is active
    , ALT_HOST_NAME     /* Name of host serving alternate path
    , ALT_HOST_TYPE     /* Type of alternate host
    , HOST_AVAIL        /* Primary host is active
    , HOST_COUNT        /* Number of paths to the device
    , HOST_NAME         /* Name of host serving the primary path
    , HOST_TYPE         /* Type of primary host (today one of "V785", "V780", "V750" or "HS50")
    , REMOTE_DEVICE     /* Device is not connected to local node
    , SERVED_DEVICE     /* Device is served to the cluster

    , SHDW_CATCHUP_COPYING /* Catch-up copy is in progress
    , SHDW_MASTER       /* Device is "virtual" master device for shadow set
    , SHDW_MASTER_NAME  /* Name of the "virtual" master device for a shadow set
    , SHDW_MEMBER       /* Device is one of the volumes making a shadow set
    , SHDW_MERGE_COPYING /* Merge copy is in progress
    , SHDW_NEXT_MBR_NAME /* Name of the next device in shadow set
/*
/**  THE FOLLOWING CODES ARE REGULAR ITEMS
/*
    , TT_PHYDEVNAM        /* Terminal physical device name - STRING - 64 bytes
    , TT_DECCRT2          /* DEC_CRT level 2 part of devdepend2 longword for
                          /* terminals.
    , MEDIA_NAME          /* Decoded media name from UCB$L_MEDIA_ID field (ie. RK07 )
    , MEDIA_TYPE          /* Decoded media type from UCB$L_MEDIA_ID field (ie. DM )
    , MEDIA_ID            /* NONdecoded media id from UCB$L_MEDIA_ID
/*
/** THE FOLLOWING CODES ARE CONTINGENCY ITEMS FOR SHADOW SUPPORT.  IF THEY
/** ARE NOT USED, THE NEXT RELEASE OF VMS CAN REUSE THESE POSITIONS FOR OTHER
/** ITEMS.  IF THEY ARE USED, A SYNONYM WITH MORE MEANING SHOULD BE DEFINED.
/*
    , SHDW_spare_bit_1          /* A spare boolean
    , SHDW_spare_bit_2          /* A spare boolean
    , SHDW_spare_string_1       /* A spare character string
    , SHDW_spare_string_2       /* A spare character string
    , SHDW_spare_integer_1      /* A spare longword integer
    , SHDW_spare_integer_2      /* A spare longword integer

/****   ADD NEW ITEM-CODES IMMEDIATELY BEFORE THIS COMMENT *****

    ) equals 2 increment 2  prefix DVI tag $;
/*
/* DVI$_item_code retrieves the item for the primary device
/* DVI$_item_code ! DVI$C_SECONDARY retrieves the item for the secondary device
/*
constant SECONDARY      equals 1  prefix DVI tag $C;    /* Get item for secondary device
/*
/* The following ACP type codes are formally defined in $AQBDEF
/* These synonyms are available to user programs and they are
/* guaranteed to be consistent by ASSUME's in SYSGETDEV.  Additions
/* to the ACP type codes in $AQBDEF should be reflected here and
/* in the ASSUMES in SYSGETDEV.
/*
constant(
      ACP_F11V1                                           /* FILES-11 STRUCTURE LEVEL 1
    , ACP_F11V2                                           /* FILES-11 STRUCTURE LEVEL 2
    , ACP_MTA                                             /* MAGTAPE
    , ACP_NET                                             /* NETWORKS
    , ACP_REM                                             /* REMOTE I/O
    , ACP_JNL                                             /* JOURNAL
    ) equals 1 increment 1  prefix DVI tag $C;

end_module $DVIDEF;
```

```
module $ERADEF;
/*
/*
/* Define erase type codes.  The codes LODUMMY and HIDUMMY are
/* used as placeholders, to make the definition of the upper and
/* lower bound erase type symbols automatic.  New erase type codes
/* should be added at the end of the list, but before HIDUMMY.
/*
/*
constant (LODUMMY,                          /*
          MEMORY,                           /* Erase main memory
          DISK,                             /* Erase disk mass storage
          TAPE,                             /* Erase magnetic tape mass storage
          HIDUMMY                           /*
         )
        equals 0 increment 1 prefix ERA tag $K;
constant MINTYPE equals ERA$K_LODUMMY+1 prefix ERA tag $K;/* Lower bound of erase type codes
constant MAXTYPE equals ERA$K_HIDUMMY-1 prefix ERA tag $K;/* Upper bound of erase type codes

end_module $ERADEF;
```

0433 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



OPCDEF
SDL

OPDEF
SDL

SCRDEF
SDL

SRMDEF
SDL

STARDEFAE
SDL

STARDEFFL
SDL

STARDEFMP
SDL

STARDEFQZ
SDL